aws

Entwicklerhandbuch

Amazon Elastic Container Service

Amazon Elastic Container Service: Entwicklerhandbuch

Table of Contents

Was ist Amazon Elastic Container Service?

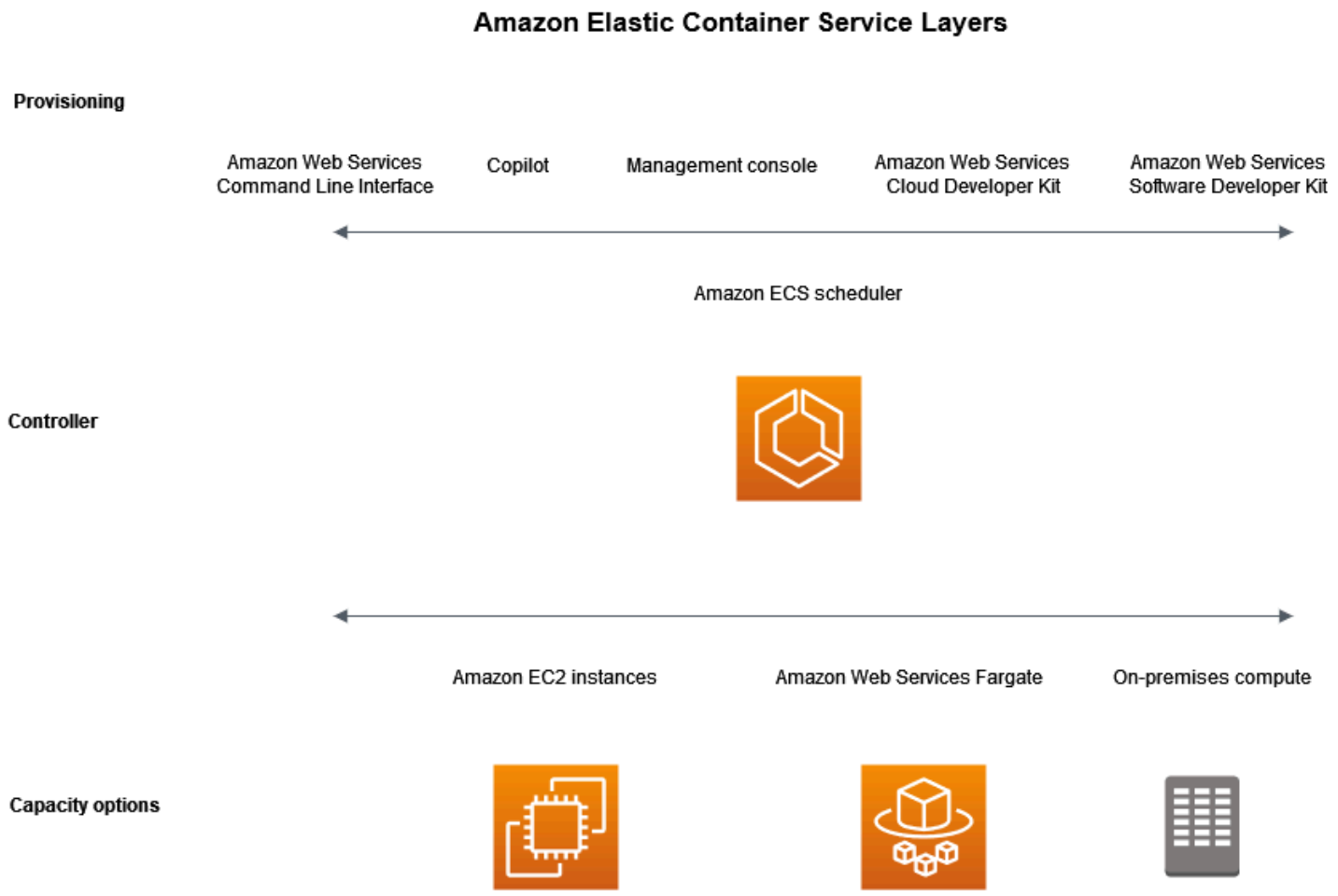
Amazon Elastic Container Service (Amazon ECS) ist ein vollständig verwalteter Container-Orchestrierungsservice, mit dem Sie containerisierte Anwendungen einfach bereitstellen, verwalten und skalieren können. Als vollständig verwalteter Service bietet Amazon ECS integrierte Best Practices für AWS Konfiguration und Betrieb. Es ist AWS sowohl in Tools von Drittanbietern als auch in Tools von Drittanbietern wie Amazon Elastic Container Registry und Docker integriert. Diese Integration erleichtert es Teams, sich auf die Erstellung der Anwendungen zu konzentrieren, nicht auf die Umgebung. Sie können Ihre Container-Workloads sowohl AWS-Regionen in der Cloud als auch vor Ort ausführen und skalieren, ohne die Komplexität der Verwaltung einer Kontrollebene.

Amazon-ECS-Terminologie und -Komponenten

Amazon ECS besteht aus drei Ebenen:

- Kapazität – Die Infrastruktur, in der Ihre Container laufen
- Controller – Stellen Ihre Anwendungen, die auf den Containern ausgeführt werden, bereit und verwalten sie
- Bereitstellung – Die Tools, die Sie als Schnittstelle zum Scheduler verwenden können, um Ihre Anwendungen und Container bereitzustellen und zu verwalten

Das folgende Diagramm zeigt die Amazon-ECS-Ebenen.



Amazon-ECS-Kapazität

Die Amazon ECS-Kapazität ist die Infrastruktur, in der Ihre Container laufen. Im Folgenden finden Sie eine Übersicht über die Kapazitätsoptionen:

- Amazon EC2 EC2-Instanzen in der Cloud AWS

  Sie wählen den Instance-Typ und die Anzahl der Instances und verwalten die Kapazität.
- Serverlos (AWS Fargate (Fargate)) in der Cloud AWS

  Fargate ist eine serverlose pay-as-you-go Compute-Engine. Mit Fargate müssen Sie keine Server verwalten, Kapazitätsplanung durchführen oder Container-Workloads aus Sicherheitsgründen isolieren.
- On-Premises-virtuelle-Maschinen (VM) oder -Server

  Amazon ECS Anywhere bietet Unterstützung für die Registrierung einer externen Instance, wie einem On-Premises-Server oder einer virtuellen Maschine (VM) in Ihren Amazon-ECS-Cluster.

Die Kapazität kann sich in einer der folgenden AWS Ressourcen befinden:

- Availability Zones
- Local Zones
- Wavelength-Zonen
- AWS-Regionen
- AWS Outposts

Amazon-ECS-Controller

Der Amazon ECS Scheduler ist die Software, die Ihre Anwendungen verwaltet.

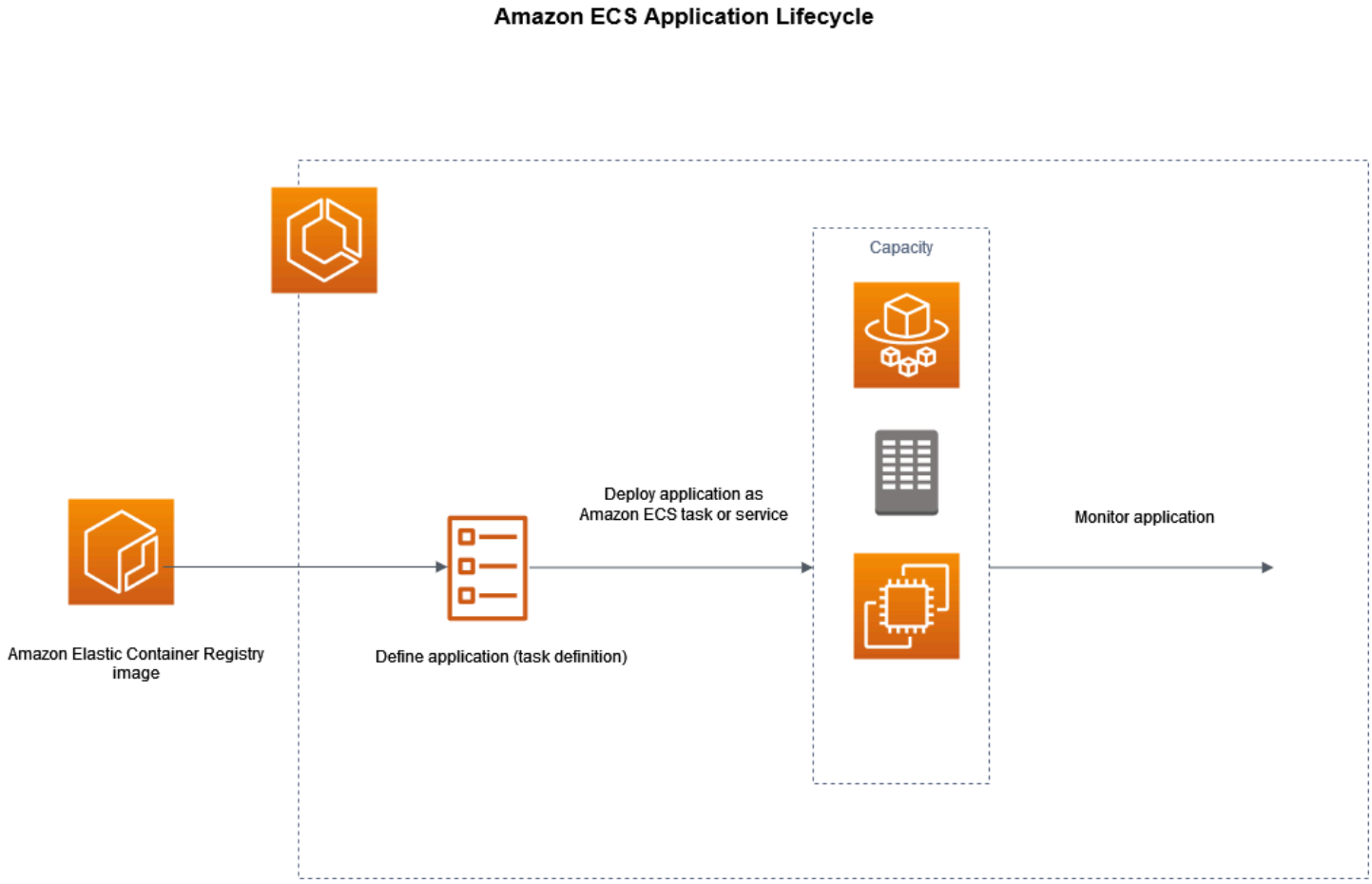
Amazon-ECS-Bereitstellung

Es gibt mehrere Optionen für die Bereitstellung von Amazon ECS:

- AWS Management Console – Bietet eine Webschnittstelle für den Zugriff auf Ihre Amazon-ECS-Ressourcen.
- AWS Command Line Interface (AWS CLI) — Stellt Befehle für eine Vielzahl von AWS Diensten bereit, darunter Amazon ECS. Es wird auf Windows, Mac und Linux unterstützt. Weitere Informationen finden Sie unter AWS Command Line Interface.
- AWS SDKs — Stellt sprachspezifische APIs bereit und kümmert sich um viele Verbindungsdetails. Dazu gehören das Berechnen von Signaturen, das Behandeln von Wiederholungsversuchen und das Behandeln von Fehlern. Weitere Informationen finden Sie unter AWS -SDKs.
- Copilot – Bietet Entwicklern ein Open-Source-Tool, mit dem sie produktionsfertige containerisierte Anwendungen auf Amazon ECS erstellen, veröffentlichen und betreiben können. Weitere Informationen finden Sie auf der Website unter Copilot. GitHub
- AWS CDK – Bietet ein Open-Source-Softwareentwicklungs-Framework, mit dem Sie Ihre Cloud-Anwendungsressourcen mithilfe vertrauter Programmiersprachen modellieren und bereitstellen können. AWS CDK stellt Ihre Ressourcen sicher und wiederholbar bereit durch AWS CloudFormation.

Anwendungslebenszyklus

Das folgende Diagramm zeigt den Anwendungslebenszyklus und wie er mit den Amazon-ECS-Komponenten funktioniert.



Sie müssen Ihre Anwendungen so gestalten, dass sie auf Containern ausgeführt werden können. Ein Container ist eine standardisierte Einheit der Softwareentwicklung, die alles enthält, was Ihre Softwareanwendung zum Ausführen benötigt. Dies beinhaltet relevanten Code, Laufzeit, Systemtools und Systembibliotheken. Container werden anhand einer schreibgeschützten Vorlage erstellt, die als Image bezeichnet wird. Images werden normalerweise aus einer Dockerdatei erstellt. Ein Dockerfile ist eine Klartextdatei, die die Anweisungen zum Erstellen eines Containers enthält. Nach der Erstellung werden diese Images in einer Registrierung wie Amazon ECR gespeichert, von wo sie heruntergeladen werden können.

Nachdem Sie Ihr Image erstellt und gespeichert haben, erstellen Sie eine Amazon-ECS-Aufgabendefinition. Eine Aufgabendefinition ist eine Vorlage für Ihre Anwendung. Es handelt sich um eine Textdatei im JSON-Format, die die Parameter und einen oder mehrere Container beschreibt, die Ihre Anwendung bilden. So können Sie beispielsweise das Image und die Parameter für das Betriebssystem angeben, welche Container verwendet werden sollen, welche Ports für Ihre Anwendung geöffnet werden sollen und welche Daten-Volumes mit den Containern in der Aufgabe verwendet werden sollen. Welche spezifischen Parameter für Ihre Aufgabendefinition verfügbar sind, hängt von den Anforderungen Ihrer spezifischen Anwendung ab.

Nachdem Sie Ihre Aufgabendefinition definiert haben, stellen Sie sie entweder als Service oder als Aufgabe in Ihrem Cluster bereit. Ein Cluster ist eine logische Gruppierung von Aufgaben oder Services, die auf der Kapazitätsinfrastruktur ausgeführt wird, die auf einem Cluster registriert ist.

Eine Aufgabe ist die Instance iierung einer Aufgabendefinition auf einer Container-Instance in einem Cluster. Sie können eine eigenständige Aufgabe ausführen oder eine Aufgabe als Teil eines Services ausführen. Sie können einen Amazon-ECS-Service verwenden, um Ihre gewünschte Anzahl von Aufgaben gleichzeitig in einem Amazon-ECS-Cluster auszuführen und zu warten. Wenn eine Ihrer Aufgaben aus irgendeinem Grund fehlschlägt oder angehalten wird, launcht der Scheduler des Amazon-ECS-Service eine andere Instance entsprechend Ihrer Aufgabendefinition. Dies geschieht, um sie zu ersetzen und dadurch Ihre gewünschte Anzahl von Aufgaben im Service beizubehalten.

Der Container-Agent wird auf jeder Container-Instance in einem Amazon-ECS-Cluster ausgeführt. Der Agent sendet Informationen über die aktuell ausgeführten Aufgaben und die Ressourcenauslastung Ihrer Container an Amazon ECS. Er startet und stoppt Aufgaben auf entsprechende Aufforderung von Amazon ECS.

Nachdem Sie die Aufgabe oder den Service bereitgestellt haben, können Sie Ihre Bereitstellung und Anwendung mithilfe eines der folgenden Tools überwachen:

- CloudWatch
- Laufzeit-Überwachung

Verwandte Informationen zu Amazon ECS

Die folgenden verwandten Ressourcen bieten Ihnen nützliche Informationen für die Arbeit mit diesem Service.

- AWS Fargate – Überblick über die Features von Fargate.
- Windows on AWS — Überblick über Windows on AWS Workloads und Dienste.
- Linux von AWS — Portfolio moderner Linux-basierter Betriebssysteme von. AWS

Tutorials für Entwickler

- AWS Compute-Blogs – Informationen über neue Features, detaillierte Einblicke in Features, Codebeispiele und bewährte Methoden.

AWS re:Post

AWS re:Post— AWS verwalteter Frage-und-Antwort-Service (Q & A), der von Experten geprüfte Antworten auf Ihre technischen Fragen per Crowdsourcing bietet.

Preisgestaltung

- Amazon-ECS-Preise – Preisübersicht für Amazon ECS.
- AWS Fargate Preisgestaltung — Preisinformationen für Fargate.

Allgemeine Ressourcen AWS

Die folgenden allgemeinen Ressourcen können Ihnen bei der Arbeit mit helfen AWS.

- Kurse und Workshops — Links zu rollen- und Spezialkursen sowie zu Übungen zum Selbststudium, mit denen Sie Ihre AWS Fähigkeiten verbessern und praktische Erfahrungen sammeln können.
- AWS Developer Center — Erkunden Sie Tutorials, laden Sie Tools herunter und erfahren Sie mehr über Veranstaltungen für Entwickler. AWS
- AWS Entwicklertools — Links zu Entwicklertools, SDKs, IDE-Toolkits und Befehlszeilentools für die Entwicklung und Verwaltung von AWS Anwendungen.
- Ressourcencenter für die ersten Schritte — Erfahren Sie, wie Sie Ihre AWS-Konto Anwendung einrichten, der AWS Community beitreten und Ihre erste Anwendung starten.
- Praktische Tutorials — Folgen Sie den step-by-step Tutorials, um Ihre erste Anwendung zu starten. AWS
- AWS Whitepapers — Links zu einer umfassenden Liste von technischen AWS Whitepapers zu Themen wie Architektur, Sicherheit und Wirtschaft, die von Solutions Architects oder anderen technischen Experten verfasst wurden. AWS
- AWS Support Center — Die zentrale Anlaufstelle für die Erstellung und Verwaltung Ihrer Fälle. AWS Support Enthält auch Links zu anderen hilfreichen Ressourcen wie Foren, häufig gestellten technischen Fragen, dem Status des Dienstes und AWS Trusted Advisor.
- AWS Support— Die wichtigste Webseite mit Informationen über AWS Support einen Support-Kanal mit schnellen Reaktionszeiten one-on-one, der Sie bei der Entwicklung und Ausführung von Anwendungen in der Cloud unterstützt.
- Kontakt – Zentraler Kontaktpunkt für Fragen zu AWS -Abrechnung, Konten, Ereignissen Missbrauch und anderen Problemen.

- AWS Nutzungsbedingungen der Website — Detaillierte Informationen zu unseren Urheberrechten und Marken, zu Ihrem Konto, Ihrer Lizenz und Ihrem Zugriff auf die Website sowie zu anderen Themen.

Erfahren Sie, wie Sie Amazon ECS-Ressourcen erstellen und verwenden

Die folgenden Anleitungen bieten eine Einführung in die Tools, die für den Zugriff auf Amazon ECS verfügbar sind, sowie in einführende Verfahren zur Ausführung von Containern. Die Docker-Grundlagen führen Sie durch die grundlegenden Schritte zum Erstellen eines Docker-Container-Images und zum Hochladen in ein privates Amazon-ECR-Repository. Die Anleitungen für die ersten Schritte führen Sie durch die Verwendung der AWS Copilot-Befehlszeilenschnittstelle und führen Sie die AWS Management Console allgemeinen Aufgaben zum Ausführen Ihrer Container auf Amazon ECS und durch. AWS Fargate

Inhalt

Einrichtung für die Verwendung von Amazon ECS

Wenn Sie bereits für Amazon Web Services (AWS) registriert sind und Amazon Elastic Compute Cloud (Amazon EC2) schon verwendet haben, müssen Sie nur wenige Schritte ausführen, um Amazon ECS nutzen zu können. Der Einrichtungsprozess für die beiden Services ist ähnlich. Der folgende Leitfaden bereitet Sie auf den Start Ihres ersten Amazon-ECS-Clusters vor.

Führen Sie zum Einrichten von Amazon ECS die folgenden Schritte aus.

Melden Sie sich an für ein AWS-Konto

Wenn Sie noch keine haben AWS-Konto, führen Sie die folgenden Schritte aus, um eine zu erstellen.

Um sich für eine anzumelden AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/signup.
2. Folgen Sie den Online-Anweisungen.

Bei der Anmeldung müssen Sie auch einen Telefonanruf entgegennehmen und einen Verifizierungscode über die Telefontasten eingeben.

Wenn Sie sich für eine anmelden AWS-Konto, Root-Benutzer des AWS-Kontoswird eine erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Aus Sicherheitsgründen sollten Sie einem Benutzer Administratorzugriff zuweisen und nur den Root-Benutzer verwenden, um Aufgaben auszuführen, für die Root-Benutzerzugriff erforderlich ist.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Sie können jederzeit Ihre aktuelle Kontoaktivität anzeigen und Ihr Konto verwalten. Rufen Sie dazu https://aws.amazon.com/ auf und klicken Sie auf Mein Konto.

Erstellen Sie einen Benutzer mit Administratorzugriff

Nachdem Sie sich für einen angemeldet haben AWS-Konto, sichern Sie Ihren Root-Benutzer des AWS-Kontos AWS IAM Identity Center, aktivieren und erstellen Sie einen Administratorbenutzer, sodass Sie den Root-Benutzer nicht für alltägliche Aufgaben verwenden.

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

1. Melden Sie sich AWS Management Consoleals Kontoinhaber an, indem Sie Root-Benutzer auswählen und Ihre AWS-Konto E-Mail-Adresse eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

   Hilfe bei der Anmeldung mit dem Root-Benutzer finden Sie unter Anmelden als Root-Benutzer im AWS-Anmeldung Benutzerhandbuch zu.
2. Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den Root-Benutzer.

   Anweisungen finden Sie unter Aktivieren eines virtuellen MFA-Geräts für Ihren AWS-Konto Root-Benutzer (Konsole) im IAM-Benutzerhandbuch.

Erstellen Sie einen Benutzer mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center Benutzerhandbuch.
2. Gewähren Sie einem Benutzer in IAM Identity Center Administratorzugriff.

Ein Tutorial zur Verwendung von IAM-Identity-Center-Verzeichnis als Identitätsquelle finden Sie unter Benutzerzugriff mit der Standardeinstellung konfigurieren IAM-Identity-Center-Verzeichnis im AWS IAM Identity Center Benutzerhandbuch.

Melden Sie sich als Benutzer mit Administratorzugriff an

- Um sich mit Ihrem IAM-Identity-Center-Benutzer anzumelden, verwenden Sie die Anmelde-URL, die an Ihre E-Mail-Adresse gesendet wurde, als Sie den IAM-Identity-Center-Benutzer erstellt haben.

  Hilfe bei der Anmeldung mit einem IAM Identity Center-Benutzer finden Sie im AWS-Anmeldung Benutzerhandbuch unter Anmeldung beim AWS Zugriffsportal.

Weisen Sie weiteren Benutzern Zugriff zu

1. Erstellen Sie in IAM Identity Center einen Berechtigungssatz, der der bewährten Methode zur Anwendung von Berechtigungen mit den geringsten Rechten folgt.

   Anweisungen finden Sie im Benutzerhandbuch unter Einen Berechtigungssatz erstellen.AWS IAM Identity Center
2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

   Anweisungen finden Sie im AWS IAM Identity Center Benutzerhandbuch unter Gruppen hinzufügen.

Erstellen einer Virtual Private Cloud

Sie können Amazon Virtual Private Cloud (Amazon VPC) verwenden, um AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk zu starten. Wir empfehlen dringend, Ihre Container-Instances in einer VPC zu starten.

Wenn Sie über eine standardmäßige VPC verfügen, können Sie diesen Abschnitt überspringen und zur nächsten Aufgabe, Eine Sicherheitsgruppe erstellen, übergehen. Informationen darüber, ob Sie über eine Standard-VPC verfügen, finden Sie unter Unterstützte Plattformen in der Amazon EC2 EC2-Konsole im Amazon EC2 EC2-Benutzerhandbuch. Andernfalls können Sie unter

Berücksichtigung der nachfolgenden Schritte eine nicht standardmäßige VPC in Ihrem Konto erstellen.

Informationen zum Erstellen einer VPC finden Sie unter Nur VPC erstellen im Amazon-VPC-Benutzerhandbuch. Verwenden Sie die folgende Tabelle, um zu bestimmen, welche Optionen auszuwählen sind.

Option	Wert
Zu erstellende Ressourcen	Nur VPC
Name	Geben Sie optional einen Namen für Ihre VPC ein.
IPv4 CIDR block (IPv4-CIDR-Block)	Manuelle IPv4-CIDR-Eingabe Die CIDR-Blockgröße muss eine Größe zwischen /16 und /28. haben.
IPv6-CIDR-Block	Kein IPv6-CIDR-Block
Tenancy	Standard

Weitere Informationen über Amazon VPC finden Sie unter Was ist Amazon VPC? im Amazon VPC-Benutzerhandbuch.

Eine Sicherheitsgruppe erstellen

Sicherheitsgruppen fungieren als Firewall für zugehörige Container-Instances. Sie kontrollieren den ein- und ausgehenden Datenverkehr auf Container-Instance-Ebene. Sie können einer Sicherheitsgruppe Regeln hinzufügen, die es Ihnen ermöglichen, von Ihrer IP-Adresse über SSH eine Verbindung zu Ihrer Container-Instance herzustellen. Sie können auch Regeln hinzufügen, die den ein- und ausgehenden HTTP- und HTTPS-Zugriff von überall zulassen. Fügen Sie alle Regeln zum Öffnen von Ports hinzu, die für Ihre Aufgaben benötigt werden. Container-Instances benötigen einen externen Netzwerkzugriff, um mit dem Amazon ECS Service-Endpunkt zu kommunizieren.

Wenn Sie planen, Container-Instances in mehreren Regionen zu starten, müssen Sie in jeder Region eine Sicherheitsgruppe erstellen. Weitere Informationen finden Sie unter Regionen und Availability Zones im Amazon EC2 EC2-Benutzerhandbuch.

> **Tip**
>
> Sie benötigen die öffentliche IP-Adresse Ihres lokalen Computers, die Sie mithilfe eines Service abrufen können. Wir stellen z. B. folgenden Service bereit: http://checkip.amazonaws.com/ oder https://checkip.amazonaws.com/. Wenn Sie einen anderen Service suchen möchten, der Ihnen Ihre IP-Adresse nennt, verwenden Sie den Suchausdruck "wie ist meine IP-Adresse". Wenn Sie eine Verbindung über einen InternetServiceanbieter (ISP) oder hinter einer Firewall ohne statische IP-Adresse herstellen, müssen Sie herausfinden, welchen IP-Adressbereich die Client-Computer verwenden.

Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Erstellen einer Sicherheitsgruppe im Amazon EC2 EC2-Benutzerhandbuch. Verwenden Sie die folgende Tabelle, um zu bestimmen, welche Optionen Sie auswählen müssen.

Option	Wert
Region	Die Region, in der Sie Ihr Schlüsselpaar erstellt haben.
Name	Ein Name, den Sie sich leicht merken können, wie etwa ecs-instances-default-cluster.
VPC	Die Standard-VPC. Sie ist mit „(default)“ (Standard) gekennzeichnet. **Note** Wenn Ihr Konto Amazon EC2 Classic unterstützt, wählen Sie die VPC aus, die

Option	Wert
	Sie in der vorherige n Aufgabe erstellt haben.

Informationen zu den ausgehenden Regeln, die Sie für Ihre Anwendungsfälle hinzufügen können, finden Sie unter Sicherheitsgruppenregeln für verschiedene Anwendungsfälle im Amazon EC2 EC2-Benutzerhandbuch.

Für Amazon-ECS-Container-Instances müssen keine eingehenden Ports geöffnet sein. Allerdings empfiehlt es sich, eine SSH-Regel hinzuzufügen, sodass Sie sich bei der Container-Instance anmelden und die Aufgaben mit Docker-Befehlen prüfen können. Sie können auch Regeln für HTTP und HTTPS hinzufügen, wenn Ihre Container-Instance eine Aufgabe hosten soll, die auf einem Webserver ausgeführt wird. Container-Instances benötigen Zugriff auf ein externes Netzwerk, um mit dem Amazon-ECS-Service-Endpunkt kommunizieren zu können. Führen Sie die folgenden Schritte aus, um diese optionale Regeln für die Sicherheitsgruppe hinzuzufügen.

Fügen Sie Ihrer Sicherheitsgruppe die folgenden drei Regeln für eingehenden Datenverkehr hinzu. Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Regeln zu Ihrer Sicherheitsgruppe hinzufügen im Amazon EC2 EC2-Benutzerhandbuch.

Option	Wert
HTTP-Regel	Typ: HTTP Quelle: Anywhere (`0.0.0.0/0`) Diese Option fügt automatis ch den IPv4-CIDR-Block 0.0.0.0/0 als Quelle hinzu. Dies ist zwar für kurze Zeit in einer Testumgebung akzeptabel, aber für Produktio nsumgebungen sehr unsicher. Autorisieren Sie in Produktio

Option	Wert	
	nsumgebungen nur eine bestimmte IP-Adresse bzw. einen bestimmten Adressber eich für den Zugriff auf Ihre Instance.	
HTTPS-Regel	Typ: HTTPS Quelle: Anywhere (`0.0.0.0/0`) Dies ist zwar für kurze Zeit in einer Testumgebung akzeptabel, aber für Produktio nsumgebungen sehr unsicher. Autorisieren Sie in Produktio nsumgebungen nur eine bestimmte IP-Adresse bzw. einen bestimmten Adressber eich für den Zugriff auf Ihre Instance.	

Option	Wert
SSH-Regel	Typ: SSH Quelle: „Custom“ (Benutzer definiert). Geben Sie die öffentliche IP-Adresse Ihres Computers oder Netzwerks in CIDR-Notation an. Um eine einzelne IP-Adresse in CIDR-Notation anzugeben, fügen Sie das Routing-Präfix `/32` hinzu. Beispiel: Wenn Ihre IP-Adresse `203.0.113.25` lautet, geben Sie `203.0.113.25/32` an. Wenn Ihr Unternehmen Adressen aus einem Bereich zuweist, geben Sie den gesamten Bereich an, z. B. `203.0.113.0/24` . **Important** Aus Sicherheitsgründen empfehlen wir, dass Sie den SSH-Zugriff nicht von allen IP-Adressen (`0.0.0.0/0`) zu Ihrer Instance erlauben, es sei denn für Testzwecke und nur für kurze Zeit.

Erstellen der Anmeldeinformationen zum Herstellen einer Verbindung mit Ihrer EC2-Instance

Für Amazon ECS wird ein Schlüsselpaar nur dann benötigt, wenn Sie den Starttyp EC2 verwenden.

AWS verwendet Public-Key-Kryptografie, um die Anmeldeinformationen für Ihre Instance zu sichern. Eine Linux-Instance, wie beispielsweise eine Amazon-ECS-Container-Instance, hat kein Passwort, das für den SSH-Zugriff verwendet werden kann. Sie verwenden ein Schlüsselpaar, um sich sicher an Ihrer Instance anzumelden. Beim Starten Ihrer Container-Instance geben Sie den Namen des Schlüsselpaares und dann bei der Anmeldung über SSH den privaten Schlüssel an.

Wenn Sie nicht bereits ein Schlüsselpaar erstellt haben, können Sie mit der Amazon EC2-Konsole eines erstellen. Wenn Sie planen, Instances in mehreren Regionen zu starten, müssen Sie in jeder Region ein Schlüsselpaar erstellen. Weitere Informationen zu Regionen finden Sie unter Regionen und Availability Zones im Amazon EC2 EC2-Benutzerhandbuch.

So erstellen Sie ein Schlüsselpaar

- Erstellen Sie ein Schlüsselpaar mit der Amazon-EC2-Konsole. Weitere Informationen zum Erstellen eines Schlüsselpaars finden Sie unter Create a key pair im Amazon EC2 EC2-Benutzerhandbuch.

Informationen darüber, wie Sie eine Verbindung zu Ihrer Instance herstellen, finden Sie unter Connect to your Linux Instance im Amazon EC2 EC2-Benutzerhandbuch.

Installieren Sie das AWS CLI

Das AWS Management Console kann verwendet werden, um alle Operationen manuell mit Amazon ECS zu verwalten. Sie können das jedoch AWS CLI auf Ihrem lokalen Desktop oder einer Entwicklerbox installieren, sodass Sie Skripts erstellen können, mit denen allgemeine Verwaltungsaufgaben in Amazon ECS automatisiert werden können.

Um das AWS CLI mit Amazon ECS zu verwenden, installieren Sie die neueste AWS CLI Version. Informationen zur Installation AWS CLI oder zum Upgrade auf die neueste Version finden Sie unter Installation der AWS Befehlszeilenschnittstelle im AWS Command Line Interface Benutzerhandbuch.

Erstellen eines Container-Images zur Verwendung auf Amazon ECS

Amazon ECS verwendet Docker-Images in Aufgabendefinitionen, um Container zu launchen. Docker ist eine Technologie, die Ihnen die Tools zum Erstellen, Ausführen, Testen und Bereitstellen verteilter Anwendungen in Containern bereitstellt.

Der Zweck der hier beschriebenen Schritte besteht darin, Sie durch das Erstellen Ihres ersten Docker-Images zu führen und dieses Image an Amazon ECR, eine Container-Registry, zu übertragen, um es in Ihren Amazon-ECS-Aufgabendefinitionen zu verwenden. Diese Anleitung setzt voraus, dass Sie ein grundlegendes Verständnis davon haben, was Docker ist und wie es funktioniert. Weitere Informationen zu Docker finden Sie unter Was ist Docker? und im Thema Docker-Übersicht.

Voraussetzungen

Bevor Sie beginnen, stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind.

- Stellen Sie sicher, dass Sie die Einrichtungsschritte für Amazon ECR abgeschlossen haben. Weitere Informationen finden Sie unter Einrichtung für Amazon ECR im Benutzerhandbuch für Amazon Elastic Container Registry.
- Ihr Benutzer verfügt über die erforderlichen IAM-Berechtigungen für den Zugriff auf den Amazon-ECR-Service und dessen Nutzung. Weitere Informationen finden Sie unter Amazon ECR-verwaltete Richtlinien.
- Sie haben Docker installiert. Informationen zu Docker-Installationsschritten für Amazon Linux 2 finden Sie unter Installation von Docker auf AL2023. Informationen zu allen anderen Betriebssystemen finden Sie in der Docker-Dokumentation unter Docker-Desktop-Übersicht.
- Sie haben das AWS CLI installiert und konfiguriert. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface im AWS Command Line Interface - Benutzerhandbuch.

Wenn Sie keine lokale Entwicklungsumgebung haben oder benötigen und Docker mithilfe einer Amazon-EC2-Instance verwenden möchten, haben wir Ihnen die folgenden Schritte bereitgestellt, um eine Amazon-EC2-Instance mit Amazon Linux 2 zu launchen und Docker Engine und die Docker-CLI zu installieren.

Installation von Docker auf AL2023

Docker ist auf vielen verschiedenen Betriebssystemen verfügbar, darunter die meisten modernen Linux-Distributionen wie Ubuntu und sogar macOS und Windows. Weitere Informationen zur Installation von Docker unter einem bestimmten Betriebssystem finden Sie im Docker-Installationshandbuch.

Für die Verwendung von Docker wird kein lokales Entwicklungssystem benötigt. Wenn Sie bereits Amazon EC2 verwenden, können Sie eine Amazon-Linux-2023-Instance starten und Docker installieren, um loszulegen.

Wenn Sie Docker bereits installiert haben, fahren Sie mit Erstellen eines Docker-Image fort.

So installieren Sie Docker auf einer Amazon-EC2-Instance mit einem Amazon-Linux-2023-AMI

1. Starten Sie eine Instance mit dem neuesten Amazon-Linux-2023-AMI. Weitere Informationen finden Sie unter Launching an Instance im Amazon EC2 EC2-Benutzerhandbuch.
2. Verbinden Sie sich mit der Instance. Weitere Informationen finden Sie unter Connect to Your Linux Instance im Amazon EC2 EC2-Benutzerhandbuch.
3. Aktualisieren Sie die installierten Pakete und den Cache der Paketverwaltung auf Ihrer Instance.

   ```
   sudo yum update -y
   ```

4. Installieren Sie das neueste Docker-Community Edition-Paket.

   ```
   sudo yum install docker
   ```

5. Starten Sie den Docker-Service.

   ```
   sudo service docker start
   ```

6. Fügen Sie den `ec2-user` zur Gruppe `docker` hinzu, sodass Sie Docker-Befehle ohne Verwendung von `sudo` ausführen können.

   ```
   sudo usermod -a -G docker ec2-user
   ```

7. Melden Sie sich ab und wieder an, um die neuen Berechtigungen der Gruppe `docker` zu übernehmen. Sie erreichen dies, indem Sie das aktuelle SSH-Terminalfenster schließen und sich über ein neues Terminalfenster wieder mit Ihrer Instance verbinden. Ihre neue SSH-Sitzung verfügt über die entsprechenden `docker`-Gruppenberechtigungen.

8. Überprüfen Sie, ob der `ec2-user` Docker-Befehle ohne `sudo` ausführen kann.

```
docker info
```

> **Note**
>
> In einigen Fällen müssen Sie möglicherweise Ihre Instance neu starten, um den `ec2-user` für den Zugriff auf den Docker-Daemon zu berechtigen. Versuchen Sie, Ihre Instance neu zu starten, wenn die folgende Fehlermeldung angezeigt wird:
>
> ```
> Cannot connect to the Docker daemon. Is the docker daemon running on this
>  host?
> ```

Erstellen eines Docker-Image

Amazon-ECS-Aufgabendefinitionen nutzen Docker-Images, um Container auf den Container-Instances in Ihren Clustern zu starten. In diesem Abschnitt erstellen Sie ein Docker-Image einer einfachen Webanwendung, testen es auf Ihrem lokalen System oder Ihrer Amazon-EC2-Instance und übertragen das Image direkt an das Container-Registry von Amazon ECR, um es in einer Amazon-ECS-Aufgabendefinition verwenden zu können.

So erstellen Sie ein Docker-Image einer einfachen Webanwendung

1. Erstellen Sie eine Datei mit dem Namen `Dockerfile`. Eine Docker-Datei ist eine Manifestdatei, die das für Ihr Docker-Image zu verwendende Basis-Image sowie die Inhalte beschreibt, die Sie darauf installieren und ausführen möchten. Weitere Informationen zu Dockerfiles finden Sie unter Dockerfile Reference.

```
touch Dockerfile
```

2. Bearbeiten Sie die soeben von Ihnen erstellte `Dockerfile` und fügen Sie die folgenden Inhalte hinzu.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Install dependencies
RUN yum update -y && \
```

```
 yum install -y httpd

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
```

Diese Docker-Datei verwendet das öffentliche Amazon-Linux-2-Image, das auf Amazon ECR Public gehostet wird. Die RUN-Anweisungen aktualisieren die Caches der Paketverwaltung, installieren einige Softwarepakete für den Webserver und schreiben dann den Inhalt "Hello World!" in das Stammverzeichnis für Dokumente des Webservers. Die EXPOSE Anweisung bedeutet, dass Port 80 auf dem Container derjenige ist, der zuhört, und die CMD Anweisung startet den Webserver.

3. Erstellen Sie das Docker-Image aus der Dockerfile.

> **Note**
>
> Einige Versionen von Docker erfordern im folgenden Befehl anstelle des unten angegebenen relativen Pfads den vollständigen Pfad zu Ihrer Dockerfile.

```
docker build -t hello-world .
```

4. Führen Sie Ihr Container-Image auf.

```
docker images --filter reference=hello-world
```

Ausgabe:

```
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
```

```
hello-world         latest              e9ffedc8c286        4 minutes ago
 194MB
```

5. Führen Sie das neu erstellte Image aus. Die Option `-p 80:80` ordnet den bereitgestellten Port 80 auf dem Container dem Port 80 auf dem Hostsystem zu. Weitere Informationen zu docker run finden Sie unter Referenz zu Docker run.

   ```
   docker run -t -i -p 80:80 hello-world
   ```

   > **Note**
   >
   > Ausgabe vom Apache-Webserver wird im Terminal-Fenster angezeigt. Sie können die Meldung „`Could not reliably determine the fully qualified domain name`“ ignorieren.

6. Öffnen Sie einen Browser und richten Sie ihn auf den Server aus, auf dem Docker ausgeführt und Ihr Container gehostet wird.

   - Wenn Sie eine EC2-Instance verwenden, ist dies der Wert für Öffentliche DNS für den Server. Dabei handelt es sich um dieselbe Adresse, mit der Sie eine Verbindung mit der Instance per SSH herstellen. Vergewissern Sie sich, dass die Sicherheitsgruppe für Ihre Instance eingehenden Datenverkehr auf Port 80 zulässt.
   - Wenn Sie Docker lokal ausführen, richten Sie Ihren Browser auf http://localhost/ aus.
   - Wenn Sie es docker-machine auf einem Windows- oder Mac-Computer verwenden, suchen Sie die IP-Adresse der VirtualBox VM, die Docker hostet, mit dem docker-machine ip Befehl und ersetzen Sie *`machine-name durch den Namen`* des Docker-Computers, den Sie verwenden.

     ```
     docker-machine ip machine-name
     ```

   Sie sollten eine Webseite mit dem Text "Hello, World!" Nachricht sehen.

7. Beenden Sie den Docker-Container, indem Sie Strg+C eingeben.

Senden Sie Ihr Image an die Amazon Elastic Container-Registry

Amazon ECR ist ein verwalteter AWS Docker-Registrierungsdienst. Sie können die Docker-Befehlszeilenschnittstelle (CLI) verwenden, um Images in Ihren Amazon ECR-Repositorys zu pushen, pullen und zu verwalten. Amazon ECR-Produktdetails, vorgestellte Fallbeispiele von Kunden und häufig gestellte Fragen finden Sie auf den Seiten mit den Amazon Elastic Container-Registry-Produktdetails.

So versehen Sie Ihr Image mit Tags und übertragen es per Push direkt zu Amazon ECR

1. Erstellen Sie ein Amazon ECR-Repository, um Ihr `hello-world`-Image zu speichern. Notieren Sie sich den `repositoryUri` in der Ausgabe.

   Ersetzen `region` Sie zum AWS-Region Beispiel durch Ihr. `us-east-1`

   ```
   aws ecr create-repository --repository-name hello-repository --region region
   ```

   Ausgabe:

   ```
   {
       "repository": {
           "registryId": "aws_account_id",
           "repositoryName": "hello-repository",
           "repositoryArn": "arn:aws:ecr:region:aws_account_id:repository/hello-
   repository",
           "createdAt": 1505337806.0,
           "repositoryUri": "aws_account_id.dkr.ecr.region.amazonaws.com/hello-
   repository"
       }
   }
   ```

2. Versehen Sie das `hello-world`-Image als Tag mit dem `repositoryUri`-Wert aus dem vorherigen Schritt.

   ```
   docker tag hello-world aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
   ```

3. Führen Sie den Befehl aws ecr get-login-password aus. Geben Sie den Registrierungs-URI an, bei dem Sie sich authentifizieren möchten. Weitere Informationen finden Sie unter Registry-Authentifizierung im Benutzerhandbuch zu Amazon-Elastic-Container-Registry.

```
aws ecr get-login-password --region region | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Ausgabe:

```
Login Succeeded
```

> **Important**
>
> Bei einem Fehler installieren oder aktualisieren Sie auf die neueste AWS CLI-Version. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface im AWS Command Line Interface -Benutzerhandbuch.

4. Senden Sie das Image per Push mit dem `repositoryUri`-Wert aus dem vorherigen Schritt zu Amazon ECR.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

Bereinigen

Um mit dem Erstellen einer Amazon-ECS-Aufgabendefinition und dem Launchen einer Aufgabe mit Ihrem Container-Image fortzufahren, gehen Sie direkt zu Nächste Schritte. Wenn Sie Ihr Amazon ECR-Image nicht mehr nutzen möchten, können Sie das Repository löschen, sodass Ihnen keine Kosten für die Image-Speicherung anfallen.

```
aws ecr delete-repository --repository-name hello-repository --region region --force
```

Nächste Schritte

Ihre Aufgabendefinitionen erfordern eine Aufgaben-Ausführungsrolle. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Nachdem Sie Ihr Container-Image erstellt und an Amazon ECR übertragen haben, können Sie dieses Image in einer Aufgabendefinition verwenden. Weitere Informationen finden Sie unter einem der folgenden Themen:

- the section called “Erfahren Sie, wie Sie eine Linux-Aufgabe für den Fargate-Starttyp erstellen”

- the section called “Erfahren Sie, wie Sie eine Windows-Aufgabe für den Fargate-Starttyp erstellen”
- Erstellen einer Amazon ECS-Linux-Aufgabe für den Fargate-Starttyp mit dem AWS CLI

Erfahren Sie, wie Sie eine Amazon ECS-Linux-Aufgabe für den Starttyp Fargate erstellen

Amazon Elastic Container Service (Amazon ECS) ist ein hoch skalierbarer, schneller Container-Management-Service, der das Ausführen, Beenden und Verwalten Ihrer Containern vereinfacht. Sie können Ihren Cluster auf einer Serverless-Infrastruktur hosten, die von Amazon ECS verwaltet wird, indem Sie Ihre Services oder Aufgaben unter AWS Fargate starten. Weitere Informationen zu Fargate finden Sie unterAWS Fargatefür Amazon ECS.

Beginnen Sie mit der Aktivierung AWS Fargate von Amazon ECS, indem Sie den Starttyp Fargate für Ihre Aufgaben in den Regionen verwenden, in denen Amazon ECS AWS Fargate unterstützt.

Führen Sie für den Einstieg in Amazon ECS auf AWS Fargate folgende Schritte aus:

Voraussetzungen

Bevor Sie beginnen, müssen Sie die Schritte unter abschließen Einrichtung für die Verwendung von Amazon ECS und sicherstellen, dass Ihr AWS Benutzer über die im `AdministratorAccess` IAM-Richtlinienbeispiel angegebenen Berechtigungen verfügt.

Die Konsole versucht, automatisch die IAM-Rolle für die Aufgabenausführung zu erstellen, die für Fargate-Aufgaben erforderlich ist. Damit die Konsole in der Lage ist, diese IAM-Rolle zu erstellen, muss eine der folgenden Bedingungen erfüllt sein:

- Ihr Benutzer hat Administratorzugriff. Weitere Informationen finden Sie unter Einrichtung für die Verwendung von Amazon ECS.
- Ihr Benutzer verfügt über die IAM-Berechtigungen zum Erstellen einer Servicerolle. Weitere Informationen finden Sie unter Eine Rolle zum Delegieren von Berechtigungen für einen AWS Dienst erstellen.
- Ein Benutzer mit Administratorzugriff hat die Rolle zur Aufgabenausführung manuell erstellt, sodass sie im Konto verfügbar ist. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Important

Für die Sicherheitsgruppe, die Sie beim Erstellen eines Services mit Ihrer Aufgabendefinition auswählen, muss Port 80 für eingehenden Datenverkehr geöffnet sein. Fügen Sie die folgende Regel für eingehenden Datenverkehr zu Ihrer Sicherheitsgruppe hinzu. Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Hinzufügen von Regeln zu Ihrer Sicherheitsgruppe im Amazon EC2 EC2-Benutzerhandbuch.

- Typ: HTTP
- Protokoll: TCP
- Portbereich: 80
- Quelle: Anywhere (`0.0.0.0/0`)

Schritt 1: Cluster erstellen

Erstellen Sie einen Cluster, der die Standard-VPC verwendet.

Bevor Sie beginnen, weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Geben Sie unter Cluster configuration (Cluster-Konfiguration) für Cluster name (Clustername) einen eindeutigen Namen ein.

   Der Name kann bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Bindestriche enthalten.
6. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).
7. (Optional) Um Ihren Cluster leichter identifizieren zu können, erweitern Sie den Bereich Tags, und konfigurieren Sie dann Ihre Tags.

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

- Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
- Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

[Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.

8. Wählen Sie Erstellen.

Schritt 2: Erstellen einer Aufgabendefinition

Eine Aufgabendefinition ist wie ein Bauplan für Ihre Anwendung. Jedes Mal, wenn Sie eine Aufgabe in Amazon ECS starten, geben Sie eine Aufgabendefinition an. Damit teilen Sie dem Service das zu verwendende Docker-Image für die Container, die Anzahl der in der Aufgabe einzusetzenden Container und die Ressourcenzuordnung für jeden Container mit.

1. Wählen Sie im Navigationsbereich Task Definitions aus.
2. Wählen Sie Create new Task Definition (Neue Aufgabendefinition erstellen), Create new revision with JSON (Neue Revision mit JSON) erstellen.
3. Kopieren Sie die folgende Beispielaufgabendefinition, fügen Sie sie in das Feld ein, und wählen Sie dann Save (Speichern).

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [
                "sh",
  "-c"
```

```
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

4. Wählen Sie Erstellen.

Schritt 3: Erstellen eines Service

Erstellen Sie einen Service mithilfe der Aufgabendefinition.

1. Klicken Sie im Navigationsbereich auf Clusters (Cluster) und wählen Sie den zu ändernden Cluster in Schritt 1: Cluster erstellen aus.
2. Wählen Sie auf der Registerkarte Services die Option Create (Erstellen) aus.
3. Unter Deployment configuration (Konfiguration der Bereitstellung), geben Sie an, wie Ihre Anwendung bereitgestellt wird.
   a. Wählen Sie unter Task definition (Aufgabendefinition) die in Schritt 2: Erstellen einer Aufgabendefinition erstellte Aufgabendefinition aus.
   b. Wählen Sie für Service name (Servicename) einen Namen für Ihren Service aus.
   c. Geben Sie für Desired tasks (Gewünschte Aufgaben) 1 ein.
4. Unter Netzwerk können Sie eine Sicherheitsgruppe erstellen oder eine bestehende Sicherheitsgruppe für Ihre Aufgabe auswählen. Stellen Sie sicher, dass in der Sicherheitsgruppe, die Sie verwenden, die Regel für eingehenden Datenverkehr unter Voraussetzungen aufgeführt ist.
5. Wählen Sie Erstellen.

Schritt 4: Anzeigen Ihres Services

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie den Cluster aus, auf dem Sie den Service ausgeführt haben.
4. Wählen Sie auf der Registerkarte Services unter Servicename den Service aus, den Sie in Schritt 3: Erstellen eines Service erstellt haben.
5. Wählen Sie die Registerkarte Aufgaben und wählen Sie dann die Aufgabe in Ihrem Service aus.
6. Wählen Sie auf der Seite Aufgabe im Abschnitt Konfiguration unter Öffentliche IP-Adresse die Option Offene Adresse aus.

Schritt 5: Bereinigen

Wenn Sie einen Amazon-ECS-Cluster nicht mehr verwenden, sollten Sie die ihm zugeordneten Ressourcen bereinigen, um Kosten für Ressourcen zu vermeiden, die Sie nicht verwenden.

Einige Amazon ECS-Ressourcen wie Aufgaben, Services, Cluster und Container-Instances werden bei Verwendung der Amazon ECS-Konsole bereinigt. Andere Ressourcen, wie Amazon EC2 EC2-Instances, Elastic Load Balancing Load Balancer und Auto Scaling Scaling-Gruppen, müssen manuell in der Amazon EC2 EC2-Konsole oder durch Löschen des AWS CloudFormation Stacks, der sie erstellt hat, bereinigt werden.

1. Klicken Sie im Navigationsbereich auf Cluster.
2. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie für dieses Tutorial erstellt haben.
3. Wählen Sie die Registerkarte Services.
4. Wählen Sie den Service und dann Löschen aus.
5. Geben Sie an der Bestätigungsaufforderung delete (löschen) ein und wählen Sie dann Delete (Löschen) aus. Alternativ können Sie die `Force delete` Option verwenden, Amazon ECS den Service in Ihrem Namen herunterskalieren zu lassen, bevor Sie ihn löschen.

   Warten Sie, bis der Service gelöscht ist.
6. Wählen Sie Delete Cluster (Cluster löschen) aus. Geben Sie an der Bestätigungsaufforderung delete ***`cluster-name`*** (cluster-name löschen) ein und wählen Sie dann Delete (Löschen) aus. Durch das Löschen des Clusters werden die zugehörigen Ressourcen bereinigt, die mit dem Cluster erstellt wurden, einschließlich Auto-Scaling-Gruppen, VPCs oder Load Balancer.

Erfahren Sie, wie Sie eine Amazon ECS-Windows-Aufgabe für den Starttyp Fargate erstellen

Beginnen Sie mit der Aktivierung AWS Fargate von Amazon ECS, indem Sie den Starttyp Fargate für Ihre Aufgaben in den Regionen verwenden, in denen Amazon ECS AWS Fargate unterstützt.

Führen Sie für den Einstieg in Amazon ECS auf AWS Fargate folgende Schritte aus:

Voraussetzungen

Bevor Sie beginnen, müssen Sie die Schritte unter abschließen Einrichtung für die Verwendung von Amazon ECS und sicherstellen, dass Ihr AWS Benutzer über die im `AdministratorAccess` IAM-Richtlinienbeispiel angegebenen Berechtigungen verfügt.

Die Konsole versucht, automatisch die IAM-Rolle für die Aufgabenausführung zu erstellen, die für Fargate-Aufgaben erforderlich ist. Damit die Konsole in der Lage ist, diese IAM-Rolle zu erstellen, muss eine der folgenden Bedingungen erfüllt sein:

- Ihr Benutzer hat Administratorzugriff. Weitere Informationen finden Sie unter Einrichtung für die Verwendung von Amazon ECS.
- Ihr Benutzer verfügt über die IAM-Berechtigungen zum Erstellen einer Servicerolle. Weitere Informationen finden Sie unter Eine Rolle zum Delegieren von Berechtigungen für einen AWS Dienst erstellen.
- Ein Benutzer mit Administratorzugriff hat die Rolle zur Aufgabenausführung manuell erstellt, sodass sie im Konto verfügbar ist. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

> **Important**
>
> Für die Sicherheitsgruppe, die Sie beim Erstellen eines Services mit Ihrer Aufgabendefinition auswählen, muss Port 80 für eingehenden Datenverkehr geöffnet sein. Fügen Sie die folgende Regel für eingehenden Datenverkehr zu Ihrer Sicherheitsgruppe hinzu. Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Hinzufügen von Regeln zu Ihrer Sicherheitsgruppe im Amazon EC2 EC2-Benutzerhandbuch.
>
> - Typ: HTTP
> - Protokoll: TCP

- Portbereich: 80
- Quelle: Anywhere (`0.0.0.0/0`)

Schritt 1: Erstellen eines Clusters

Sie können einen neuen Cluster namens windows erstellen, der die Standard-VPC verwendet.

Um einen Cluster mit dem zu erstellen AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Geben Sie unter Cluster configuration (Cluster-Konfiguration) für Cluster name (Clustername) windows ein.
6. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).
7. (Optional) Um Ihren Cluster leichter identifizieren zu können, erweitern Sie den Bereich Tags, und konfigurieren Sie dann Ihre Tags.

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

   [Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.
8. Wählen Sie Erstellen.

Schritt 2: Registrieren einer Windows-Aufgabendefinition

Bevor Sie in Ihrem Amazon-ECS-Cluster Windows-Container ausführen können, müssen Sie eine Aufgabendefinition registrieren. Das folgende Beispiel einer Aufgabendefinition zeigt eine einfache

Webseite auf Port 8080 einer Container-Instance mit dem `mcr.microsoft.com/windows/servercore/iis`-Container-Image.

Um die Beispiel-Aufgabendefinition bei der zu registrieren AWS Management Console

1. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
2. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
3. Kopieren Sie die folgende Beispielaufgabendefinition, fügen Sie sie in das Feld ein, und wählen Sie dann Save (Speichern).

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
```

```
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

4. Überprüfen Sie Ihre Informationen und wählen Sie Erstellen.

Schritt 3: Erstellen eines Services mit Ihrer Aufgabendefinition

Nachdem Sie Ihre Aufgabendefinition registriert haben, können Sie damit Aufgaben in Ihrem Cluster platzieren. Mit der folgenden Vorgehensweise wird ein Service mit Ihrer Aufgabendefinition erstellt und eine Aufgabe in Ihrem Cluster platziert.

So erstellen Sie mit der Konsole einen Service aus der Aufgabendefinition

1. Klicken Sie im Navigationsbereich auf Clusters (Cluster) und wählen Sie den zu ändernden Cluster in Schritt 1: Erstellen eines Clusters aus.
2. Wählen Sie auf der Registerkarte Services die Option Create (Erstellen) aus.
3. Unter Deployment configuration (Konfiguration der Bereitstellung), geben Sie an, wie Ihre Anwendung bereitgestellt wird.
    a. Wählen Sie unter Task definition (Aufgabendefinition) die in Schritt 2: Registrieren einer Windows-Aufgabendefinition erstellte Aufgabendefinition aus.
    b. Wählen Sie für Service name (Servicename) einen Namen für Ihren Service aus.
    c. Geben Sie für Desired tasks (Gewünschte Aufgaben) 1 ein.
4. Unter Netzwerk können Sie eine Sicherheitsgruppe erstellen oder eine bestehende Gruppe auswählen. Stellen Sie sicher, dass in der Sicherheitsgruppe, die Sie verwenden, die Regel für eingehenden Datenverkehr unter Voraussetzungen aufgeführt ist.
5. Wählen Sie Erstellen.

Schritt 4: Anzeigen Ihres Services

Nachdem Ihr Service eine Aufgabe im Cluster gestartet hat, können Sie den Service anzeigen und die IIS-Testseite in einem Browser öffnen, um zu prüfen, ob der Container ausgeführt wird.

> **Note**
>
> Es kann bis zu 15 Minuten dauern, um Ihre Container-Instance herunterzuladen und die Windows-Container-Basisebenen zu extrahieren.

So zeigen Sie Ihren Service an

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie den Cluster aus, auf dem Sie den Service ausgeführt haben.
4. Wählen Sie auf der Registerkarte Services unter Servicename den Service aus, den Sie in Schritt 3: Erstellen eines Services mit Ihrer Aufgabendefinition erstellt haben.
5. Wählen Sie die Registerkarte Aufgaben und wählen Sie dann die Aufgabe in Ihrem Service aus.
6. Wählen Sie auf der Seite Aufgabe im Abschnitt Konfiguration unter Öffentliche IP-Adresse die Option Offene Adresse aus.

Schritt 5: Bereinigen

Wenn Sie einen Amazon-ECS-Cluster nicht mehr verwenden, sollten Sie die ihm zugeordneten Ressourcen bereinigen, um Kosten für Ressourcen zu vermeiden, die Sie nicht verwenden.

Einige Amazon ECS-Ressourcen wie Aufgaben, Services, Cluster und Container-Instances werden bei Verwendung der Amazon ECS-Konsole bereinigt. Andere Ressourcen, wie Amazon EC2 EC2-Instances, Elastic Load Balancing Load Balancer und Auto Scaling Scaling-Gruppen, müssen manuell in der Amazon EC2 EC2-Konsole oder durch Löschen des AWS CloudFormation Stacks, der sie erstellt hat, bereinigt werden.

1. Klicken Sie im Navigationsbereich auf Cluster.
2. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie für dieses Tutorial erstellt haben.
3. Wählen Sie die Registerkarte Services.
4. Wählen Sie den Service und dann Löschen aus.
5. Geben Sie an der Bestätigungsaufforderung delete (löschen) ein und wählen Sie dann Delete (Löschen) aus.

   Warten Sie, bis der Service gelöscht ist.

6. Wählen Sie Delete Cluster (Cluster löschen) aus. Geben Sie an der Bestätigungsaufforderung delete ***`cluster-name`*** (cluster-name löschen) ein und wählen Sie dann Delete (Löschen) aus. Durch das Löschen des Clusters werden die zugehörigen Ressourcen bereinigt, die mit dem Cluster erstellt wurden, einschließlich Auto-Scaling-Gruppen, VPCs oder Load Balancer.

Erfahren Sie, wie Sie eine Amazon ECS-Windows-Aufgabe für den EC2-Starttyp erstellen

Beginnen Sie mit Amazon ECS mithilfe des EC2-Starttyps, indem Sie eine Aufgabendefinition registrieren, einen Cluster erstellen und einen Service in der Konsole erstellen.

Führen Sie die folgenden Schritte aus, um mit Amazon ECS mit dem Starttyp EC2 zu beginnen.

Voraussetzungen

Bevor Sie beginnen, müssen Sie die Schritte unter abschließen Einrichtung für die Verwendung von Amazon ECS und sicherstellen, dass Ihr AWS Benutzer über die im `AdministratorAccess` IAM-Richtlinienbeispiel angegebenen Berechtigungen verfügt.

Die Konsole versucht, automatisch die IAM-Rolle für die Aufgabenausführung zu erstellen, die für Fargate-Aufgaben erforderlich ist. Damit die Konsole in der Lage ist, diese IAM-Rolle zu erstellen, muss eine der folgenden Bedingungen erfüllt sein:

- Ihr Benutzer hat Administratorzugriff. Weitere Informationen finden Sie unter Einrichtung für die Verwendung von Amazon ECS.
- Ihr Benutzer verfügt über die IAM-Berechtigungen zum Erstellen einer Servicerolle. Weitere Informationen finden Sie unter Eine Rolle zum Delegieren von Berechtigungen für einen AWS Dienst erstellen.
- Ein Benutzer mit Administratorzugriff hat die Rolle zur Aufgabenausführung manuell erstellt, sodass sie im Konto verfügbar ist. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

> **Important**
>
> Für die Sicherheitsgruppe, die Sie beim Erstellen eines Services mit Ihrer Aufgabendefinition auswählen, muss Port 80 für eingehenden Datenverkehr geöffnet sein. Fügen Sie die folgende Regel für eingehenden Datenverkehr zu Ihrer Sicherheitsgruppe hinzu.

Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Hinzufügen von Regeln zu Ihrer Sicherheitsgruppe im Amazon EC2 EC2-Benutzerhandbuch.

- Typ: HTTP
- Protokoll: TCP
- Portbereich: 80
- Quelle: Anywhere (`0.0.0.0/0`)

Schritt 1: Erstellen eines Clusters

Ein Amazon-ECS-Cluster ist eine logische Gruppierung von Aufgaben, Services und Container-Instances.

Die folgenden Schritte führen Sie durch das Erstellen eines Clusters mit einer registrierten Amazon-EC2-Instance, die es uns ermöglicht, eine Aufgabe darauf auszuführen. Wenn ein bestimmtes Feld nicht erwähnt wird, belassen Sie die Standardwerte der Konsole.

So erstellen Sie einen neuen Cluster (Amazon ECS-Konsole)

Bevor Sie beginnen, weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Geben Sie unter Cluster configuration (Cluster-Konfiguration) für Cluster name (Clustername) einen eindeutigen Namen ein.

   Der Name kann bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Bindestriche enthalten.
6. (Optional) Um die VPC und Subnetze zu ändern, in denen Ihre Aufgaben und Services gelauncht werden, führen Sie unter Networking (Netzwerk) einen der folgenden Vorgänge aus:

   - Um ein Subnetz zu entfernen, wählen Sie unter Subnets (Subnetze) X für jedes Subnetz, das Sie entfernen möchten.

- Um zu einer anderen VPC als der Standard-VPC zu wechseln, wählen Sie unter VPC eine vorhandene VPC und dann unter Subnets (Subnetze) jedes Subnetz aus.

7. Um Ihrem Cluster Amazon-EC2-Instances hinzuzufügen, erweitern Sie Infrastruktur und wählen Sie dann Amazon-EC2-Instances aus. Konfigurieren Sie als Nächstes die Auto-Scaling-Gruppe, die als Kapazitätsanbieter fungiert:

   a. Um eine vorhandene Auto-Scaling-Gruppe zu verwenden, wählen Sie die Gruppe aus der Auto-Scaling-Gruppe (ASG) aus.

   b. Um eine Auto-Scaling-Gruppe zu erstellen, wählen Sie in der Auto-Scaling-Gruppe (ASG) Create new group (Neue Gruppe erstellen) und geben Sie dann die folgenden Details zur Gruppe an:

      - Wählen Sie für Operating system/Architecture (Betriebssystem/Architektur) das Amazon-ECS-optimierte AMI für die Auto-Scaling-Gruppen-Instances aus.
      - Wählen Sie für EC2 instance type (EC2-Instance-Typ) den Instance-Typ für Ihre Workloads aus. Weitere Informationen zu den verschiedenen Instance-Typen finden Sie auf Amazon-EC2-Instances.

        Die verwaltete Skalierung funktioniert am besten, wenn Ihre Auto-Scaling-Gruppe dieselben oder ähnliche Instance-Typen verwendet.
      - Wählen Sie für SSH key pair (SSH-Schlüsselpaar) das Paar aus, das Ihre Identität nachweist, wenn Sie eine Verbindung zur Instance herstellen.
      - Geben Sie für Capacity (Kapazität) die minimale Anzahl und die maximale Anzahl von Instances ein, die in der Auto-Scaling-Gruppe gelauncht werden sollen. Amazon EC2 EC2-Instances fallen Kosten an, solange sie in Ihren AWS Ressourcen vorhanden sind. Weitere Informationen dazu finden Sie unter Amazon EC2 – Preise.

8. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).

9. (Optional) Um die Cluster-Tags zu verwalten, erweitern Sie Tags und führen Sie dann eine der folgenden Vorgänge aus:

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

[Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.

10. Wählen Sie Erstellen.

Schritt 2: Registrieren einer Aufgabendefinition

Um die Beispiel-Aufgabendefinition bei der zu registrieren AWS Management Console

1. Wählen Sie im Navigationsbereich Task Definitions aus.
2. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
3. Kopieren Sie die folgende Beispielaufgabendefinition, fügen Sie sie in das Feld ein, und wählen Sie dann Speichern.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
```

```
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
}
```

4. Überprüfen Sie Ihre Informationen und wählen Sie Erstellen.

Schritt 3: Erstellen eines Service

Ein Amazon-ECS-Service hilft Ihnen beim gleichzeitigen Ausführen und Aufrechterhalten einer festgelegten Anzahl von Instances einer Aufgabendefinition in einem Amazon-ECS-Cluster. Wenn eine Ihrer Aufgaben aus irgendeinem Grund fehlschlägt oder angehalten wird, startet der Amazon-ECS-Service Scheduler eine andere Instance Ihrer Aufgabendefinition, um die Aufgabe zu ersetzen und die gewünschte Anzahl von Aufgaben im Service aufrechtzuerhalten. Weitere Informationen zu Services finden Sie unter Amazon-ECS-Dienstleistungen.

So erstellen Sie einen Service

1. Klicken Sie im Navigationsbereich auf Cluster.
2. Wählen Sie den Cluster aus, den Sie in Schritt 1: Erstellen eines Clusters erstellt haben.
3. Wählen Sie auf der Registerkarte Services die Option Create (Erstellen).
4. Führen Sie im Abschnitt Environment (Umgebung) die folgenden Schritte aus:
    a. Wählen Sie für Compute options (Berechnungsoptionen) den Starttyp aus.
    b. Als Launch type (Starttyp) wählen Sie EC2 aus
5. Führen Sie im Abschnitt Deployment configuration (Gewünschte Konfiguration) Folgendes aus:
    a. Wählen Sie unter Family (Familie) die in Schritt 2: Registrieren einer Aufgabendefinition erstellte Aufgabendefinition aus.
    b. Wählen Sie für Service name (Servicename) einen Namen für Ihren Service aus.
    c. Geben Sie für Desired tasks (Gewünschte Aufgaben) 1 ein.
6. Überprüfen Sie die Optionen, und wählen Sie Erstellen.

7. Wählen Sie View Service (Service anzeigen), um Ihren Service zu überprüfen.

Schritt 4: Anzeigen Ihres Service

Der Service ist eine webbasierte Anwendung, mit der Sie die Container mit einem Webbrowser anzeigen können.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie den Cluster aus, auf dem Sie den Service ausgeführt haben.
4. Wählen Sie auf der Registerkarte Services unter Servicename den Service aus, den Sie in Schritt 3: Erstellen eines Service erstellt haben.
5. Wählen Sie die Registerkarte Aufgaben und wählen Sie dann die Aufgabe in Ihrem Service aus.
6. Wählen Sie auf der Seite Aufgabe im Abschnitt Konfiguration unter Öffentliche IP-Adresse die Option Offene Adresse aus. Der folgende Screenshot zeigt die erwartete Ausgabe.

Schritt 5: Bereinigen

Wenn Sie einen Amazon-ECS-Cluster nicht mehr verwenden, sollten Sie die ihm zugeordneten Ressourcen bereinigen, um Kosten für Ressourcen zu vermeiden, die Sie nicht verwenden.

Einige Amazon ECS-Ressourcen wie Aufgaben, Services, Cluster und Container-Instances werden bei Verwendung der Amazon ECS-Konsole bereinigt. Andere Ressourcen, wie Amazon EC2 EC2-Instances, Elastic Load Balancing Load Balancer und Auto Scaling Scaling-Gruppen, müssen manuell in der Amazon EC2 EC2-Konsole oder durch Löschen des AWS CloudFormation Stacks, der sie erstellt hat, bereinigt werden.

1. Klicken Sie im Navigationsbereich auf Cluster.
2. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie für dieses Tutorial erstellt haben.
3. Wählen Sie die Registerkarte Services.
4. Wählen Sie den Service und dann Löschen aus.
5. Geben Sie an der Bestätigungsaufforderung delete (löschen) ein und wählen Sie dann Delete (Löschen) aus.

   Warten Sie, bis der Service gelöscht ist.
6. Wählen Sie Delete Cluster (Cluster löschen) aus. Geben Sie an der Bestätigungsaufforderung delete ***cluster-name*** (cluster-name löschen) ein und wählen Sie dann Delete (Löschen) aus. Durch das Löschen des Clusters werden die zugehörigen Ressourcen bereinigt, die mit dem Cluster erstellt wurden, einschließlich Auto-Scaling-Gruppen, VPCs oder Load Balancer.

Amazon ECS-Entwicklertools – Übersicht

Unabhängig davon, ob Sie Teil eines großen Unternehmens oder eines Startups sind, Amazon ECS bietet eine Vielzahl von Tools, mit denen Sie Ihre Container schnell in Betrieb nehmen können, unabhängig von Ihrem Fachwissen. Sie können Amazon ECS wie folgt nutzen:

- Erfahren Sie mehr über Ihre Containeranwendungen und -dienste, entwickeln, verwalten und visualisieren Sie diese mithilfe von AWS Management Console.
- Führen Sie bestimmte Aktionen für Amazon ECS-Ressourcen mit automatisierten Bereitstellungen durch Programmierung oder Skripts mithilfe von AWS Command Line Interface, AWS SDKs oder der ECS-API aus.
- Definieren und verwalten Sie alle AWS Ressourcen in Ihrer Umgebung mit automatisierter Bereitstellung mithilfe vonAWS CloudFormation.
- Verwenden Sie den vollständigen AWS Copilot CLI end-to-end Entwickler-Workflow, um Containeranwendungen zu erstellen, zu veröffentlichen und zu betreiben, die den AWS Best Practices für die Infrastruktur entsprechen.
- Definieren Sie mithilfe Ihrer bevorzugten Programmiersprache Infrastruktur oder Architektur als Code mit AWS CDK.
- Sie können Anwendungen, die On-Premises oder auf Amazon-EC2-Instances oder in beiden gehostet werden, mithilfe des AWS App2Container-integriertes Portabilitäts- und Tooling-Ökosystem für Container verwenden.
- Stellen Sie eine Anwendung auf Amazon ECS bereit oder testen Sie lokale Container mit Containern, die in Amazon ECS mithilfe des Docker-Compose-Dateiformats mit dem Amazon ECS-CLI ausgeführt werden.
- Starten von Containern aus Docker-Desktop-Integration mit Amazon ECS mithilfe von Amazon ECS in Docker Desktop.

AWS Management Console

Das AWS Management Console ist eine browserbasierte Oberfläche für die Verwaltung von Amazon ECS-Ressourcen. Die Konsole bietet einen visuellen Überblick über den Service und erleichtert die Erkundung der Features und Funktionen von Amazon ECS, ohne dass zusätzliche Tools verwendet werden müssen. Es stehen viele verwandte Tutorials und exemplarische Vorgehensweisen zur Verfügung, die Sie durch die Verwendung der Konsole führen können.

Ein Tutorial zur Verwendung der Konsole finden Sie unter Erfahren Sie, wie Sie Amazon ECS-Ressourcen erstellen und verwenden.

Zu Beginn bevorzugen viele Kunden die Verwendung der Konsole, da sie sofort ein visuelles Feedback darüber bietet, ob die von ihnen ergriffenen Maßnahmen erfolgreich waren. AWS Kunden, die mit dem vertraut sind AWS Management Console, können verwandte Ressourcen wie Load Balancer und Amazon EC2 EC2-Instances problemlos verwalten.

Beginnen Sie mit dem. AWS Management Console

AWS Command Line Interface

Das AWS Command Line Interface (AWS CLI) ist ein einheitliches Tool, mit dem Sie Ihre AWS Dienste verwalten können. Mit diesem einen Tool allein können Sie sowohl mehrere AWS Dienste steuern als auch diese Dienste mithilfe von Skripten automatisieren. Die Amazon ECS-Befehle in der AWS CLI sind ein Spiegelbild der Amazon ECS-API.

AWS bietet zwei Gruppen von Befehlszeilentools: die AWS Command Line Interface(AWS CLI) und die AWS Tools for Windows PowerShell. Weitere Informationen finden Sie im AWS Command Line Interface -Benutzerhandbuch und im AWS Tools for Windows PowerShell -Benutzerhandbuch.

Das AWS CLI ist für Kunden geeignet, die Skripterstellung und Schnittstellen mit einem Befehlszeilentool bevorzugen und daran gewöhnt sind und genau wissen, welche Aktionen sie auf ihren Amazon ECS-Ressourcen ausführen möchten. Dies AWS CLI ist auch hilfreich für Kunden, die sich mit den Amazon ECS-APIs vertraut machen möchten. Kunden können den verwenden AWS CLI , um eine Reihe von Vorgängen auf Amazon ECS-Ressourcen auszuführen, darunter Erstellungs-, Lese-, Aktualisierungs- und Löschvorgänge, direkt von der Befehlszeilenschnittstelle aus.

Verwenden Sie die, AWS CLI wenn Sie mit den Amazon ECS-APIs und den entsprechenden CLI-Befehlen vertraut sind oder sich damit vertraut machen möchten und automatisierte Skripts schreiben und bestimmte Aktionen auf Amazon ECS-Ressourcen ausführen möchten.

AWS CloudFormation

AWS CloudFormation und Terraform für Amazon ECS bieten Ihnen leistungsstarke Möglichkeiten, Ihre Infrastruktur als Code zu definieren. Sie können ganz einfach verfolgen, welche Version Ihrer Vorlage oder Ihres AWS CloudFormation -Stacks zu jeder Zeit ausgeführt wird und bei Bedarf auf eine vorherige Version zurückgesetzt wird. Sie können Infrastruktur- und

Anwendungsbereitstellungen auf die gleiche automatisierte Weise durchführen. Diese Flexibilität und Automatisierung machen AWS CloudFormation Terraform zu zwei beliebten Formaten für die Bereitstellung von Workloads aus Continuous Delivery-Pipelines in Amazon ECS.

Weitere Informationen zu finden Sie unter. AWS CloudFormationErstellen von Amazon ECS-Ressourcen mit AWS CloudFormation

Verwenden Sie AWS CloudFormation oder Terraform, wenn Sie Infrastrukturbereitstellungen und Anwendungen auf Amazon ECS automatisieren und alle AWS Ressourcen in Ihrer Umgebung explizit definieren und verwalten möchten.

AWS Copilot CLI

Die AWS Copilot CLI (Command Line Interface) ist ein umfassendes Tool, mit dem Kunden in Containern und Umgebungen verpackte Anwendungen auf Amazon ECS direkt aus ihrem Quellcode bereitstellen und betreiben können. Wenn Sie AWS Copilot verwenden, können Sie diese Operationen ohne Verständnis von Amazon ECS-Elementen wie Application Load Balancers, öffentlichen Subnetzen, Aufgaben, Diensten AWS und Clustern ausführen. AWS Copilot erstellt in Ihrem Namen AWS Ressourcen auf der Grundlage eigenwilliger Servicemuster, wie z. B. einem Webservice mit Lastenausgleich oder einem Backend-Service, und bietet so eine unmittelbare Produktionsumgebung für containerisierte Anwendungen. Sie können die Bereitstellung über eine AWS CodePipeline Pipeline in mehreren Umgebungen, Konten oder Regionen durchführen, die alle innerhalb der CLI verwaltet werden können. Mit AWS Copilot können Sie auch Bedieneraufgaben ausführen, wie z. B. das Anzeigen von Protokollen und den Zustand Ihres Dienstes. AWS Copilot ist ein all-in-one Tool, mit dem Sie Ihre Cloud-Ressourcen einfacher verwalten können, sodass Sie sich auf die Entwicklung und Verwaltung Ihrer Anwendungen konzentrieren können.

Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.

Verwenden Sie den vollständigen end-to-end Entwickler-Workflow von AWS Copilot, um Containeranwendungen zu erstellen, zu veröffentlichen und zu betreiben, die den AWS Best Practices für die Infrastruktur entsprechen.

AWS CDK

Das AWS Cloud Development Kit (AWS CDK) ist ein Open-Source-Framework für die Softwareentwicklung, mit dem Sie Ihre Cloud-Anwendungsressourcen mithilfe vertrauter

Programmiersprachen modellieren und bereitstellen können. AWS CDK stellt Ihre Ressourcen auf sichere und wiederholbare Weise bereit. AWS CloudFormation Mit dem CDK können Kunden ihre Umgebung mit weniger Codezeilen in derselben Sprache generieren, die sie zum Erstellen ihrer Anwendung verwendet haben. Amazon ECS stellt ein Modul im CDK bereit, das `ecs-patterns` benannt ist und gemeinsame Architekturen erstellt. Ein verfügbares Muster ist `ApplicationLoadBalancedFargateService()`. Dieses Muster erstellt einen Cluster, eine Aufgabendefinition und zusätzliche Ressourcen, um einen Amazon ECS-Dienst mit Lastenausgleich auf AWS Fargate auszuführen.

Weitere Informationen finden Sie unter Erstellen von Amazon ECS-Ressourcen mit dem AWS CDK.

Verwenden Sie den, AWS CDK wenn Sie Infrastruktur oder Architektur als Code in Ihrer bevorzugten Programmiersprache definieren möchten. Sie können z. B. die gleiche Sprache verwenden, die Sie zum Schreiben Ihrer Anwendungen verwenden.

AWS App2Container

Manchmal verfügen Unternehmenskunden möglicherweise bereits über Anwendungen, die On-Premises oder auf EC2-Instances oder beidem gehostet werden. Sie interessieren sich für die Portabilität und das Tooling-Ökosystem von Containern speziell auf Amazon ECS und müssen zunächst eine Containerisierung durchführen. AWS App2Container ermöglicht es Ihnen, genau das zu tun. App2Container (A2C) ist ein Befehlszeilentool zur Modernisierung von .NET- und Java-Anwendungen in containerisierte Anwendungen. A2C analysiert und erstellt eine Bestandsaufnahme aller Anwendungen, die in virtuellen Maschinen, On-Premises oder in der Cloud ausgeführt werden. Nachdem Sie die Anwendung ausgewählt haben, die Sie containerisieren möchten, packt A2C das Anwendungsartefakt und identifizierte Abhängigkeiten in Container-Images. Anschließend werden die Netzwerkports konfiguriert und die Amazon ECS-Aufgabe generiert. Schließlich wird eine CloudFormation Vorlage erstellt, die Sie bei Bedarf bereitstellen oder ändern können.

Weitere Informationen finden Sie unter Erste Schritte mit AWS -App2Container.

Verwenden Sie App2Container, wenn Sie Anwendungen haben, die On-Premises oder auf Amazon-EC2-Instances oder beidem gehostet werden.

Amazon ECS-CLI

Die Amazon ECS-CLI ermöglicht es Ihnen, Ihre Anwendungen auf Amazon ECS und AWS Fargate unter Verwendung des Docker Compose-Dateiformats auszuführen. Sie können Ressourcen schnell

bereitstellen, Images mithilfe von Amazon ECR pushen und pullen und ausgeführte Anwendungen auf Amazon ECS oder AWS Fargateüberwachen. Sie können auch Container testen, die lokal zusammen mit Containern in der Cloud innerhalb der CLI ausgeführt werden.

Weitere Informationen finden Sie unter Erste Schritte mit der Amazon ECS-Befehlszeilenschnittstelle.

Verwenden Sie die ECS-CLI, wenn Sie über eine Compose Anwendung verfügen und diese auf Amazon ECS bereitstellen möchten, oder testen Sie lokale Container mit Containern, die in Amazon ECS in der Cloud ausgeführt werden.

Docker-Desktop-Integration mit Amazon ECS

AWS und Docker haben zusammengearbeitet, um eine vereinfachte Entwicklerumgebung zu schaffen, mit der Sie Container auf Amazon ECS direkt mithilfe von Docker-Tools bereitstellen und verwalten können. Sie können Ihre Container jetzt lokal mit Docker Desktop und Docker Compose erstellen und testen und sie dann in Amazon ECS on Fargate bereitstellen. Laden Sie Docker Desktop herunter und registrieren Sie sich optional für eine Docker-ID, um mit der Amazon ECS und Docker-Integration zu beginnen. Weitere Informationen finden Sie unter Docker-Desktop und Docker-ID-Anmeldung.

Anfänger von Containern lernen oft Container mithilfe von Docker-Tools wie Docker CLI und Docker Compose kennen. Dies macht die Verwendung des Docker Compose CLI-Plug-ins für Amazon ECS zu einem natürlichen nächsten Schritt beim Ausführen von Containern AWS nach dem lokalen Testen. Docker bietet eine exemplarische Vorgehensweise zum Bereitstellen von Containern auf Amazon ECS. Weitere Informationen finden Sie unter Docker Compose CLI — Amazon ECS.

Sie können zusätzliche Amazon ECS-Funktionen wie Service Discovery, Load Balancing und andere AWS Ressourcen für die Verwendung mit ihren Anwendungen mit Docker Desktop nutzen.

Sie können das Docker Compose CLI-Plugin für Amazon ECS auch direkt von GitHub herunterladen. Weitere Informationen finden Sie unter Docker Compose CLI Plugin für Amazon ECS auf GitHub.

AWS SDKs

Sie können AWS SDKs auch verwenden, um Amazon ECS-Ressourcen und -Operationen in einer Vielzahl von Programmiersprachen zu verwalten. Die SDKs stellen Module bereit, um Aufgaben zu erledigen, einschließlich Aufgaben in der folgenden Liste.

- Kryptographisches Signieren Ihrer Serviceanfragen

- Wiederholen von Anfragen
- Umgang mit Fehlerreaktionen

Weitere Informationen über verfügbare SDKs finden Sie unter Tools für Amazon Web Services.

Übersicht

Mit den vielen Optionen zur Auswahl können Sie die Optionen auswählen, die am besten für Sie geeignet sind. Berücksichtigen Sie dabei die folgenden Optionen.

- Wenn Sie visuell orientiert sind, können Sie Container visuell erstellen und bedienen, indem Sie die AWS Management Console verwenden.
- Wenn Sie CLIs bevorzugen, sollten Sie AWS Copilot oder das verwenden. AWS CLI Wenn Sie das Docker-Ökosystem bevorzugen, können Sie auch die Funktionen von ECS innerhalb der Docker-CLI nutzen, um zu AWS bereitzustellen. Nachdem diese Ressourcen bereitgestellt wurden, können Sie sie über die CLI oder visuell über die Konsole verwalten.
- Wenn Sie ein Entwickler sind, können Sie den verwenden, AWS CDK um Ihre Infrastruktur in derselben Sprache wie Ihre Anwendung zu definieren. Sie können CDK und AWS Copilot verwenden, um in CloudFormation Vorlagen zu exportieren, wo Sie detaillierte Einstellungen ändern, andere AWS Ressourcen hinzufügen und Bereitstellungen mithilfe von Skripten oder einer CI/CD-Pipeline automatisieren können, z. B. AWS CodePipeline

Die AWS CLI SDKs oder die ECS-API sind nützliche Tools zur Automatisierung von Aktionen auf ECS-Ressourcen und eignen sich daher ideal für die Bereitstellung. Für die Bereitstellung von Anwendungen können AWS CloudFormation Sie eine Vielzahl von Programmiersprachen oder eine einfache Textdatei verwenden, um alle für Ihre Anwendungen benötigten Ressourcen zu modellieren und bereitzustellen. Anschließend können Sie Ihre Anwendung in mehreren Regionen und Konten automatisiert und sicher bereitstellen. Sie können beispielsweise Ihren ECS-Cluster, Ihre Dienste, Aufgabendefinitionen oder Kapazitätsanbieter als Code in einer Datei definieren und mithilfe der AWS CLI CloudFormation Befehle bereitstellen.

Um Betriebsaufgaben auszuführen, können Sie Ressourcen mithilfe der SDK- oder ECS-API programmgesteuert anzeigen und verwalten. AWS CLI Befehle wie `describe-tasks` oder `list-services` zeigen die neuesten Metadaten oder eine Liste aller Ressourcen an. Ähnlich wie bei Bereitstellungen können Kunden eine Automatisierung schreiben, die Befehle wie `update-service` enthalten, um Korrekturmaßnahmen bei der Erkennung einer Ressource bereitzustellen, die

unerwartet gestoppt wurde. Sie können Ihre Dienste auch mit Copilot betreiben. AWS Befehle wie `copilot svc logs` oder `copilot app show` bieten Details zu jedem Ihrer Microservices oder zu Ihrer Anwendung als Ganzes an.

Kunden können alle verfügbaren Werkzeuge verwenden, die in diesem Dokument erwähnt werden und sie in verschiedenen Kombinationen verwenden. ECS Tooling bietet verschiedene Wege, um bestimmte Tools zu absolvieren, um andere zu verwenden, die Ihren sich ändernden Anforderungen entsprechen. Beispielsweise können Sie sich für eine detailliertere Steuerung von Ressourcen entscheiden oder bei Bedarf mehr Automatisierung entscheiden. Darüber hinaus bietet ECS eine große Auswahl an Tools für eine breite Palette von Anforderungen und Fachkenntnissen.

Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen

Die Befehle der AWS Copilot-Befehlszeilenschnittstelle (CLI) vereinfachen die Erstellung, Veröffentlichung und den Betrieb produktionsreifer containerisierter Anwendungen auf Amazon ECS von einer lokalen Entwicklungsumgebung aus. Die AWS Copilot CLI ist auf Entwickler-Workflows abgestimmt, die bewährte Methoden für moderne Anwendungen unterstützen: von der Nutzung der Infrastruktur als Code bis hin zur Erstellung einer CI/CD-Pipeline, die im Namen eines Benutzers bereitgestellt wird. Verwenden Sie die AWS Copilot-CLI als Teil Ihres täglichen Entwicklungs- und Testzyklus als Alternative zum. AWS Management Console

AWS Copilot unterstützt derzeit Linux-, MacOS- und Windows-Systeme. Weitere Informationen zur neuesten Version der AWS Copilot-CLI finden Sie unter Releases.

> **Note**
>
> Der Quellcode für die AWS Copilot-CLI ist verfügbar unter. GitHub Wir empfehlen Ihnen, uns eventuelle Belange und Änderungswünsche mitzuteilen. Amazon Web Services bietet derzeit jedoch keine Unterstützung für die Ausführung modifizierter Kopien des AWS -Copilot-Codes. Melden Sie Probleme mit AWS Copilot, indem Sie uns auf Gitter kontaktieren oder GitHubdort, wo Sie Probleme melden, Feedback geben und Fehler melden können.

Hinweise zur Installation der AWS Copilot-CLI finden Sie unter. Installation der AWS Copilot-CLI Hinweise zur Bereitstellung einer Beispiel-App finden Sie unter. Bereitstellung einer Amazon ECS-Beispielanwendung mithilfe der AWS Copilot-CLI Zusätzliche Dokumentation für die AWS Copilot-CLI ist auf der AWS Copilot-Website verfügbar.

Installation der AWS Copilot-CLI

Sie können die AWS Copilot-CLI mithilfe von Homebrew oder durch manuelles Herunterladen der Binärdatei mit den folgenden Schritten installieren.

Verwenden Sie Homebrew

Der folgende Befehl wird verwendet, um die AWS Copilot-CLI mit Homebrew auf Ihrem macOS- oder Linux-System zu installieren. Vor der Installation sollten Sie Homebrew installiert haben. Weitere Informationen finden Sie unter Homebrew.

```
brew install aws/tap/copilot-cli
```

Binärdatei herunterladen

Als Alternative zu Homebrew können Sie die AWS Copilot-CLI manuell auf Ihrem macOS-, Windows- oder Linux-System installieren. Verwenden Sie den folgenden Befehl für Ihr Betriebssystem, um die Binärdatei herunterzuladen. Die macOS- und Linux-Beispiele enthalten auch Befehle, die Ausführungsberechtigungen auf die Binärdatei anwenden, und führen das Hilfemenü auf, um zu überprüfen, ob die Installation funktioniert.

macOS

Für macOS:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Für macOS-ARM-Systeme:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Linux

Für Linux-x86-Systeme (64-Bit):

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Für Linux-ARM-Systeme:

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Windows

Mit PowerShell können Sie den folgenden Befehl ausführen:

```
New-Item -Path 'C:\copilot' -ItemType directory; `
  Invoke-WebRequest -OutFile 'C:\copilot\copilot.exe' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe
```

(Optional) Überprüfen der manuell installierten AWS -Copilot-CLI mit PGP-Signaturen

Die ausführbaren AWS Copilot-CLI-Dateien werden mithilfe von PGP-Signaturen kryptografisch signiert. Die PGP-Signaturen können verwendet werden, um die Gültigkeit der ausführbaren AWS Copilot-CLI-Datei zu überprüfen. Verwenden Sie die folgenden Schritte, um die Signaturen mithilfe des GnuPG-Tools zu überprüfen.

1. Laden Sie GnuPG herunter und installieren Sie es. Weitere Informationen finden Sie auf der GnuPG Website.

   macOS

   Wir empfehlen den Einsatz von Homebrew. Installieren Sie Homebrew mithilfe der Anweisungen von ihrer Website. Weitere Informationen finden Sie unter Homebrew. Nachdem Homebrew installiert ist, verwenden Sie den folgenden Befehl aus Ihrem macOS-Terminal:

   ```
   brew install gnupg
   ```

Linux

Installieren Sie gpg mit dem Paket-Manager auf Ihrer Linux-Variante.

Windows

Laden Sie das einfache Windows-Installationsprogramm von der GnuPG-Website herunter und installieren Sie es als Administrator. Nachdem Sie GnuPG installiert haben, schließen Sie den Administrator und öffnen Sie ihn erneut. PowerShell

Weitere Informationen finden Sie unter GnuPG-Download.

2. Stellen Sie sicher, dass der GnuPG-Pfad Ihrem Umgebungspfad hinzugefügt wurde.

macOS

```
echo $PATH
```

Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

```
PATH=$PATH:<path to GnuPG executable files>
```

Linux

```
echo $PATH
```

Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

3. Erstellen Sie eine lokale Nur-Text-Datei.

   macOS

   Geben Sie im Terminal Folgendes ein:

   ```
   touch <public_key_filename.txt>
   ```

   Öffnen Sie die Datei mit. TextEdit

   Linux

   Erstellen Sie eine Textdatei in einem Texteditor wie gedit. Speichern als `public_key_filename.txt`

   Windows

   Erstellen Sie eine Textdatei in einem Texteditor wie Notepad. Speichern als `public_key_filename.txt`

4. Fügen Sie den folgenden Inhalt des öffentlichen Amazon-ECS-PGP-Schlüssels hinzu und speichern Sie die Datei.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
```

```
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
```

```
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
```

```
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```

Die Details des öffentlichen Amazon ECS-PGP-Schlüssels als Referenz:

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

5. Importieren Sie die Datei mit dem den öffentlichen Amazon-ECS-PGP-Schlüssel mit dem folgenden Befehl im Terminal.

```
gpg --import <public_key_filename.txt>
```

6. Laden Sie die AWS Copilot-CLI-Signaturen herunter. Die Signaturen sind ASCII-getrennte PGP-Signaturen, die in Dateien mit der Erweiterung `.asc` gespeichert sind. Die Signaturen-Datei hat denselben Namen wie die entsprechende ausführbare Datei, mit dem Zusatz `.asc`.

macOS

Verwenden Sie für macOS-Systeme den folgenden Befehl.

```
sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-darwin.asc
```

Linux

Führen Sie für Linux-x86 (64-Bit)-Systeme den folgenden Befehl aus.

```
sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-linux.asc
```

Führen Sie für Linux-ARM-Systeme den folgenden Befehl aus.

```
sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-linux-arm64.asc
```

Windows

Führen Sie mit PowerShell den folgenden Befehl aus:

```
Invoke-WebRequest -OutFile 'C:\copilot\copilot.asc' https://github.com/aws/copilot-cli/releases/latest/download/copilot-windows.exe.asc
```

7. Überprüfen Sie die Signatur mit folgendem Befehl:

   - Für macOS- und Linux-Systeme:

     ```
     gpg --verify copilot.asc /usr/local/bin/copilot
     ```

   - Für Windows-Systeme:

     ```
     gpg --verify 'C:\copilot\copilot.asc' 'C:\copilot\copilot.exe'
     ```

   Erwartete Ausgabe:

   ```
   gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
   ```

```
gpg:                using RSA key DE3CBD61ADAF8B8E
gpg: Good signature from "Amazon ECS <ecs-security@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: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
```

> **Important**
>
> Die Warnung in der Ausgabe wird erwartet und ist unproblematisch. Sie tritt auf, weil keine Vertrauenskette zwischen Ihrem persönlichen PGP-Schlüssel (falls Sie einen haben) und dem Amazon ECS-PGP-Schlüssel vorhanden ist. Weitere Informationen finden Sie unter Web of trust (Netz des Vertrauens).

8. Führen Sie bei Windows-Installationen den folgenden Befehl auf Powershell aus, um das AWS Copilot-Verzeichnis zum Pfad hinzuzufügen.

```
$Env:PATH += ";<path to Copilot executable files>"
```

Bereitstellung einer Amazon ECS-Beispielanwendung mithilfe der AWS Copilot-CLI

Nach der Installation der AWS Copilot-CLI können Sie die folgenden Schritte ausführen, um eine Beispiel-App bereitzustellen, die Bereitstellung zu überprüfen und Ressourcen zu bereinigen.

Voraussetzungen

Überprüfen Sie zu Beginn, ob die folgenden Voraussetzungen erfüllt sind:

- Installieren und Konfigurieren der AWS CLI. Weitere Informationen finden Sie unter AWS - Befehlszeilenschnittstelle.
- Führen Sie aus`aws configure`, um ein Standardprofil einzurichten, das die AWS Copilot-CLI zur Verwaltung Ihrer Anwendung und Dienste verwendet.
- Installieren und Ausführen von Docker. Weitere Informationen finden Sie unter Erste Schritte mit Docker.

Stellen Sie eine Amazon ECS-Beispielanwendung mit einem einzigen Befehl bereit

1. Stellen Sie mit dem folgenden Befehl eine Beispiel-Webanwendung bereit, die aus einem GitHub Repository geklont wurde. Weitere Informationen zu AWS Copilot `init` und seinen Flags finden Sie in der AWS Copilot-Dokumentation.

```
git clone https://github.com/aws-samples/aws-copilot-sample-service.git demo-app &&
 \
cd demo-app &&                                \
copilot init --app demo                       \
  --name api                                  \
  --type 'Load Balanced Web Service'          \
  --dockerfile './Dockerfile'                 \
  --port 80                                   \
  --deploy
```

2. Nach Abschluss der Bereitstellung gibt die AWS Copilot-CLI eine URL zurück, mit der Sie die Bereitstellung überprüfen können. Sie können auch die folgenden Befehle verwenden, um den Status der App zu überprüfen.

   - Listet alle Ihre AWS Copilot-Anwendungen auf.

     ```
     copilot app ls
     ```

   - Zeigen Sie Informationen zu den Umgebungen und Diensten in Ihrer Anwendung an.

     ```
     copilot app show
     ```

   - Zeigen Sie Informationen zu Ihren Umgebungen an.

     ```
     copilot env ls
     ```

   - Zeigen Sie Informationen über den Service an, einschließlich Endpunkte, Kapazität und zugehöriger Ressourcen.

     ```
     copilot svc show
     ```

   - Liste aller Dienste in einer Anwendung.

     ```
     copilot svc ls
     ```

- Protokolle eines bereitgestellten Dienstes anzeigen.

```
copilot svc logs
```

- Servicestatus anzeigen.

```
copilot svc status
```

3. Wenn Sie mit dieser Demo fertig sind, führen Sie den folgenden Befehl aus, um die zugehörigen Ressourcen zu bereinigen und zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen.

```
copilot app delete
```

Erstellen von Amazon ECS-Ressourcen mit dem AWS CDK

Das AWS Cloud Development Kit (AWS CDK) ist ein Infrastructure-as-Code (IAC) -Framework, mit dem Sie die AWS Cloud-Infrastruktur mithilfe einer Programmiersprache Ihrer Wahl definieren können. Um Ihre eigene Cloud-Infrastruktur zu definieren, schreiben Sie zunächst eine App (in einer der unterstützten Sprachen des CDK), die einen oder mehrere Stacks enthält. Anschließend synthetisieren Sie es zu einer AWS CloudFormation Vorlage und stellen Ihre Ressourcen auf Ihrem bereit. AWS-Konto Folgen Sie den Schritten in diesem Thema, um einen containerisierten Webserver mit Amazon Elastic Container Service (Amazon ECS) und dem AWS CDK auf Fargate bereitzustellen.

Die AWS Construct-Bibliothek, die im CDK enthalten ist, stellt Module bereit, mit denen Sie die bereitgestellten Ressourcen modellieren können. AWS-Services Für die gängigen Dienste stellt die Bibliothek kuratierte Konstrukte bereit, die intelligente Standards und bewährte Methoden bereitstellen. Eines dieser Module, insbesondere `aws-ecs-patterns`, bietet Abstraktionen auf hoher Ebene, mit denen Sie Ihren containerisierten Service und alle notwendigen unterstützenden Ressourcen in nur wenigen Codezeilen definieren können.

In diesem Thema wird das Konstrukt `ApplicationLoadBalancedFargateService` verwendet. Dieses Konstrukt stellt einen Amazon-ECS-Service auf Fargate hinter einem Application Load Balancer bereit. Das `aws-ecs-patterns`-Modul enthält auch Konstrukte, die ein Network Load Balancer verwenden und auf Amazon EC2 ausgeführt werden.

Bevor Sie mit dieser Aufgabe beginnen, richten Sie Ihre AWS CDK Entwicklungsumgebung ein und installieren Sie die, AWS CDK indem Sie den folgenden Befehl ausführen. Anweisungen zum Einrichten Ihrer AWS CDK Entwicklungsumgebung finden Sie unter Erste Schritte mit den AWS CDK — Voraussetzungen.

```
npm install -g aws-cdk
```

> **Note**
>
> Bei diesen Anweisungen wird davon ausgegangen, dass Sie AWS CDK Version 2 verwenden.

Themen

Schritt 1: Richten Sie Ihr AWS CDK Projekt ein

Erstellen Sie ein Verzeichnis für Ihre neue AWS CDK App und initialisieren Sie das Projekt.

TypeScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language typescript
```

JavaScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language javascript
```

Python

```
mkdir hello-ecs
cd hello-ecs
cdk init --language python
```

Nachdem das Projekt gestartet wurde, aktivieren Sie die virtuelle Umgebung des Projekts und installieren Sie die AWS CDK Basisabhängigkeiten.

```
source .venv/bin/activate
python -m pip install -r requirements.txt
```

Java

```
mkdir hello-ecs
cd hello-ecs
cdk init --language java
```

Importieren Sie dieses Maven-Projekt in Ihre Java-IDE. Verwenden Sie in Eclipse beispielsweise Datei >Import >Maven >Vorhandene Maven-Projekte.

C#

```
mkdir hello-ecs
cd hello-ecs
cdk init --language csharp
```

Go

```
mkdir hello-ecs
cd hello-ecs
cdk init --language go
```

> **Note**
>
> Die AWS CDK Anwendungsvorlage verwendet den Namen des Projektverzeichnisses, um Namen für Quelldateien und Klassen zu generieren. In diesem Beispiel trägt das Verzeichnis den Namen `hello-ecs`. Wenn Sie einen anderen Projektverzeichnisnamen verwenden, stimmt Ihre App nicht mit diesen Anweisungen überein.

AWS CDK v2 enthält stabile Konstrukte für alle AWS-Services in einem einzigen Paket, das aufgerufen wird`aws-cdk-lib`. Dieses Paket wird als Abhängigkeit installiert, wenn Sie das Projekt initialisieren. Wenn Sie mit bestimmten Programmiersprachen arbeiten, wird das Paket installiert, wenn Sie das Projekt zum ersten Mal erstellen. Dieses Thema behandelt, wie wir ein Konstrukt von Amazon ECS Patterns, das hochrangige Abstraktionen für die Arbeit mit Amazon ECS bietet, verwenden. Dieses Modul stützt sich auf Amazon-ECS-Konstrukte und andere Konstrukte, um die für Ihre Amazon-ECS-Anwendung erforderlichen Ressourcen bereitzustellen.

Die Namen, die Sie zum Importieren dieser Bibliotheken in Ihre CDK-Anwendung verwenden, unterscheiden sich möglicherweise geringfügig, je nachdem welche Programmiersprache Sie verwenden. Als Referenz finden Sie nachfolgend die Namen, die in jeder unterstützten CDK-Programmiersprache verwendet werden.

TypeScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

JavaScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

Python

```
aws_cdk.aws_ecs
aws_cdk.aws_ecs_patterns
```

Java

```
software.amazon.awscdk.services.ecs
software.amazon.awscdk.services.ecs.patterns
```

C#

```
Amazon.CDK.AWS.ECS
Amazon.CDK.AWS.ECS.Patterns
```

Go

```
github.com/aws/aws-cdk-go/awscdk/v2/awsecs
github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns
```

Schritt 2: Verwenden Sie den AWS CDK , um einen containerisierten Webserver auf Fargate zu definieren

Verwenden Sie das Container-Image von. `amazon-ecs-sample` DockerHub Dieses Image enthält eine PHP Web-App, die unter Amazon Linux 2 läuft.

Bearbeiten Sie in dem von Ihnen erstellten AWS CDK Projekt die Datei, die die Stack-Definition enthält, so, dass sie einem der folgenden Beispiele ähnelt.

> **Note**
>
> Ein Stack ist eine Einheit der Bereitstellung. Alle Ressourcen müssen sich in einem Stack befinden, und alle Ressourcen, die sich in einem Stack befinden, werden gleichzeitig bereitgestellt. Wenn eine Ressource nicht bereitgestellt werden kann, werden alle anderen, die bereits bereitgestellt wurden, zurückgesetzt. Eine AWS CDK App kann mehrere Stapel enthalten, und Ressourcen in einem Stapel können auf Ressourcen in einem anderen Stapel verweisen.

TypeScript

Aktualisieren Sie `lib/hello-ecs-stack.ts`, so dass es Folgendem entspricht.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as ecs from 'aws-cdk-lib/aws-ecs';
import * as ecsp from 'aws-cdk-lib/aws-ecs-patterns';

export class HelloEcsStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
```

```
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
      },
      publicLoadBalancer: true
    });
  }
}
```

JavaScript

Aktualisieren Sie `lib/hello-ecs-stack.js`, so dass es Folgendem entspricht.

```
const cdk = require('aws-cdk-lib');
const { Construct } = require('constructs');
const ecs = require('aws-cdk-lib/aws-ecs');
const ecsp = require('aws-cdk-lib/aws-ecs-patterns');

class HelloEcsStack extends cdk.Stack {
  constructor(scope = Construct, id = string, props = cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
      },
      publicLoadBalancer: true
    });
  }
}

module.exports = { HelloEcsStack }
```

Python

Aktualisieren Sie `hello-ecs/hello_ecs_stack.py`, so dass es Folgendem entspricht.

```
import aws_cdk as cdk
from constructs import Construct

import aws_cdk.aws_ecs as ecs
import aws_cdk.aws_ecs_patterns as ecsp

class HelloEcsStack(cdk.Stack):
```

```
    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        ecsp.ApplicationLoadBalancedFargateService(self, "MyWebServer",
            task_image_options=ecsp.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            public_load_balancer=True
        )
```

Java

Aktualisieren Sie `src/main/java/com.myorg/HelloEcsStack.java`, so dass es Folgendem entspricht.

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.ecs.ContainerImage;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedFargateService;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedTaskImageOptions;

public class HelloEcsStack extends Stack {
    public HelloEcsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloEcsStack(final Construct scope, final String id, final StackProps
 props) {
        super(scope, id, props);

        ApplicationLoadBalancedFargateService.Builder.create(this, "MyWebServer")
          .taskImageOptions(ApplicationLoadBalancedTaskImageOptions.builder()
            .image(ContainerImage.fromRegistry("amazon/amazon-ecs-sample"))
            .build())
          .publicLoadBalancer(true)
          .build();
    }
}
```

C#

Aktualisieren Sie `src/HelloEcs/HelloEcsStack.cs`, so dass es Folgendem entspricht.

```
using Amazon.CDK;
using Constructs;
using Amazon.CDK.AWS.ECS;
using Amazon.CDK.AWS.ECS.Patterns;
namespace HelloEcs
{
    public class HelloEcsStack : Stack
    {
        internal HelloEcsStack(Construct scope, string id, IStackProps props =
 null) : base(scope, id, props)
        {
            new ApplicationLoadBalancedFargateService(this, "MyWebServer",
                new ApplicationLoadBalancedFargateServiceProps
                {
                    TaskImageOptions = new ApplicationLoadBalancedTaskImageOptions
                    {
                        Image = ContainerImage.FromRegistry("amazon/amazon-ecs-
sample")
                    },
                    PublicLoadBalancer = true
                });
        }
    }
}
```

Go

Aktualisieren Sie `hello-ecs.go`, so dass es Folgendem entspricht.

```
package main

import (
 "github.com/aws/aws-cdk-go/awscdk/v2"
 // "github.com/aws/aws-cdk-go/awscdk/v2/awssqs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns"
 "github.com/aws/constructs-go/constructs/v10"
 "github.com/aws/jsii-runtime-go"
)
```

```
type HelloEcsStackProps struct {
 awscdk.StackProps
}

func NewHelloEcsStack(scope constructs.Construct, id string, props
 *HelloEcsStackProps) awscdk.Stack {
 var sprops awscdk.StackProps
 if props != nil {
  sprops = props.StackProps
 }
 stack := awscdk.NewStack(scope, &id, &sprops)

 // The code that defines your stack goes here

 // example resource
 // queue := awssqs.NewQueue(stack, jsii.String("HelloEcsQueue"),
 &awssqs.QueueProps{
 //  VisibilityTimeout: awscdk.Duration_Seconds(jsii.Number(300)),
 // })
 res := awsecspatterns.NewApplicationLoadBalancedFargateService(stack,
 jsii.String("MyWebServer"),
  &awsecspatterns.ApplicationLoadBalancedFargateServiceProps{
   TaskImageOptions: &awsecspatterns.ApplicationLoadBalancedTaskImageOptions{
    Image: awsecs.ContainerImage_FromRegistry(jsii.String("amazon/amazon-ecs-
sample"), &awsecs.RepositoryImageProps{}),
   },
  },
 )
 awscdk.NewCfnOutput(stack, jsii.String("LoadBalancerDNS"),
 &awscdk.CfnOutputProps{Value: res.LoadBalancer().LoadBalancerDnsName()})

 return stack
}

func main() {
 defer jsii.Close()

 app := awscdk.NewApp(nil)

 NewHelloEcsStack(app, "HelloEcsStack", &HelloEcsStackProps{
  awscdk.StackProps{
   Env: env(),
  },
 })
```

```
 app.Synth(nil)
}

// env determines the AWS environment (account+region) in which our stack is to
// be deployed. For more information see: https://docs.aws.amazon.com/cdk/latest/
guide/environments.html
func env() *awscdk.Environment {
 // If unspecified, this stack will be "environment-agnostic".
 // Account/Region-dependent features and context lookups will not work, but a
 // single synthesized template can be deployed anywhere.
 //---------------------------------------------------------------------------
 return nil

 // Uncomment if you know exactly what account and region you want to deploy
 // the stack to. This is the recommendation for production stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String("123456789012"),
 //  Region:  jsii.String("us-east-1"),
 // }

 // Uncomment to specialize this stack for the AWS Account and Region that are
 // implied by the current CLI configuration. This is recommended for dev
 // stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String(os.Getenv("CDK_DEFAULT_ACCOUNT")),
 //  Region:  jsii.String(os.Getenv("CDK_DEFAULT_REGION")),
 // }
}
```

Das vorangegangene kurze Snippet beinhaltet Folgendes:

- Der logische Name des Dienstes: `MyWebServer`.
- Das Container-Image, das abgerufen wurde von DockerHub:`amazon/amazon-ecs-sample`.
- Andere relevante Informationen, wie die Tatsache, dass der Load Balancer eine öffentliche Adresse hat und über das Internet zugänglich ist.

Das erstellt AWS CDK alle Ressourcen, die für die Bereitstellung des Webservers erforderlich sind, einschließlich der folgenden Ressourcen. Diese Ressourcen wurden in diesem Beispiel weggelassen.

- Amazon-ECS-Cluster
- Amazon VPC- und Amazon EC2-Instances
- Auto Scaling-Gruppe
- Application Load Balancer
- IAM-Rollen und -Richtlinien

Einige automatisch bereitgestellte Ressourcen werden von allen Amazon-ECS-Diensten, die im Stack definiert sind, geteilt.

Speichern Sie die Quelldatei, und führen Sie dann das Kommando `cdk synth` im Hauptverzeichnis Ihrer App aus. Der AWS CDK führt die App aus, synthetisiert AWS CloudFormation daraus eine Vorlage und zeigt dann die Vorlage an. Die Vorlage ist eine etwa 600-zeilige YAML-Datei. Der Anfang der Datei wird hier gezeigt. Ihre Vorlage kann sich von diesem Beispiel unterscheiden.

```
Resources:
  MyWebServerLB3B5FD3AB:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      LoadBalancerAttributes:
        - Key: deletion_protection.enabled
          Value: "false"
      Scheme: internet-facing
      SecurityGroups:
        - Fn::GetAtt:
            - MyWebServerLBSecurityGroup01B285AA
            - GroupId
      Subnets:
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1Subnet3C273B99
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2Subnet95FF715A
      Type: application
    DependsOn:
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1DefaultRouteFF4E2178
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2DefaultRouteB1375520
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/Resource
```

```
  MyWebServerLBSecurityGroup01B285AA:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Automatically created Security Group for ELB
 HelloEcsStackMyWebServerLB06757F57
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: Allow from anyone on port 80
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId:
        Ref: EcsDefaultClusterMnL3mNNYNVpc7788A521
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/SecurityGroup/Resource
# and so on for another few hundred lines
```

Um den Dienst in Ihrem bereitzustellen AWS-Konto, führen Sie den `cdk deploy` Befehl im Hauptverzeichnis Ihrer Anwendung aus. Sie werden aufgefordert, die von Ihnen AWS CDK generierten IAM-Richtlinien zu genehmigen.

Die Bereitstellung dauert mehrere Minuten, in denen mehrere Ressourcen AWS CDK erstellt werden. Die letzten Zeilen der Ausgabe der Bereitstellung enthalten den öffentlichen Hostnamen des Load Balancers und Ihre URL für den neuen Webserver. Diese sind Folgende:

```
Outputs:
HelloEcsStack.MyWebServerLoadBalancerDNSXXXXXXX = Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
HelloEcsStack.MyWebServerServiceURLYYYYYYYY = http://Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
```

Schritt 3: Test des Webservers

Kopieren Sie die URL aus der Bereitstellungsausgabe und fügen Sie diese in Ihren Webbrowser ein. Die folgende Willkommensnachricht vom Webserver wird angezeigt.

Schritt 4: Bereinigen

Wenn Sie mit dem Webserver fertig sind, beenden Sie den Dienst mit dem CDK, indem Sie das Kommando `cdk destroy` im Hauptverzeichnis Ihrer Anwendung ausführen. Auf diese Weise wird verhindert, dass Ihnen zukünftig unbeabsichtigte Gebühren anfallen.

Nächste Schritte

Weitere Informationen zur Entwicklung einer AWS Infrastruktur mithilfe von finden Sie im AWS CDK Entwicklerhandbuch. AWS CDK

Informationen zum Schreiben von AWS CDK Apps in der Sprache Ihrer Wahl finden Sie im Folgenden:

TypeScript

- Arbeiten mit AWS CDK dem TypeScript

JavaScript

- Mit der Tinte AWS CDK arbeiten JavaScript

Python

- Arbeiten mit dem AWS CDK in Python

Java

- Arbeiten mit dem AWS CDK in Java

C#

- Arbeiten mit dem AWS CDK in C#

Go

Arbeiten mit dem AWS CDK in Go

Weitere Informationen zu den in diesem Thema verwendeten AWS Construct Library-Modulen finden Sie in den folgenden AWS CDK API-Referenzübersichten.

- aws-ecs
- aws-ecs-patterns

Erstellen von Amazon ECS-Ressourcen mit AWS CloudFormation

Amazon ECS ist integriert AWS CloudFormation, ein Service, mit dem Sie AWS Ressourcen mit von Ihnen definierten Vorlagen modellieren und einrichten können. Auf diese Weise können Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen. Mithilfe von AWS CloudFormation können Sie eine Vorlage erstellen, die alle gewünschten AWS Ressourcen beschreibt, z. B. bestimmte Amazon ECS-Cluster. Und AWS CloudFormation übernimmt dann die Bereitstellung und Konfiguration dieser Ressourcen für Sie.

Wenn Sie sie verwenden AWS CloudFormation, können Sie Ihre Vorlage wiederverwenden, um Ihre Amazon ECS-Ressourcen konsistent und wiederholbar einzurichten. Sie beschreiben Ihre Ressourcen einmal und stellen dann dieselben Ressourcen erneut für mehrere AWS-Konten und AWS-Regionen bereit.

AWS CloudFormation Vorlagen

Um Ressourcen für Amazon ECS und verwandte Services bereitzustellen und zu konfigurieren, stellen Sie sicher, dass Sie mit AWS CloudFormation Vorlagen vertraut sind. AWS CloudFormation Vorlagen sind Textdateien im JSON- oder YAML-Format, die die Ressourcen beschreiben, die Sie in Ihren AWS CloudFormation Stacks bereitstellen möchten. Wenn Sie mit dem JSON- oder YAML-Format oder mit beiden nicht vertraut sind, können Sie AWS CloudFormation Designer verwenden, um mit der Verwendung von Vorlagen zu beginnen. AWS CloudFormation Weitere Informationen finden Sie unter Was ist AWS CloudFormation Designer? im AWS CloudFormation Benutzerhandbuch.

Amazon ECS unterstützt das Erstellen von Clustern, Aufgabendefinitionen, Services und Aufgabensets in AWS CloudFormation. Die folgenden Beispiele zeigen, wie Sie Ressourcen mit diesen Vorlagen erstellen können, indem Sie AWS CLI verwenden. Sie können diese Ressourcen

auch mithilfe der AWS CloudFormation -Konsole erstellen. Weitere Informationen zum Erstellen von Ressourcen über die AWS CloudFormation -Konsole finden Sie im AWS CloudFormation Benutzerhandbuch.

Beispielvorlagen

Erstellen von Amazon ECS-Ressourcen über separate Stapel

Die folgenden Beispiele zeigen, wie Sie Amazon ECS-Ressourcen erstellen können, indem Sie separate Stapel für jede Ressource verwenden.

Aufgabendefinitionen

Sie können die folgende Vorlage verwenden, um eine Fargate Linux-Aufgabe zu erstellen.

JSON

```
  {
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
         "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
                            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS
 Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": "httpd:2.4",
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "awslogs-group": "/ecs/fargate-task-definition",
```

```
                            "awslogs-region": "us-east-1",
                            "awslogs-stream-prefix": "ecs"
                        }
                    },
                    "Name": "sample-fargate-app",
                    "PortMappings": [
                        {
                            "ContainerPort": 80,
                            "HostPort": 80,
                            "Protocol": "tcp"
                        }
                    ]
                }
            ],
            "Cpu": 256,
            "ExecutionRoleArn": "arn:aws:iam::aws_account_id:role/
ecsTaskExecutionRole",
            "Family": "task-definition-cfn",
            "Memory": 512,
            "NetworkMode": "awsvpc",
            "RequiresCompatibilities": [
                "FARGATE"
            ],
            "RuntimePlatform": {
                "OperatingSystemFamily": "LINUX"
            }
        }

    }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSTaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
```

```
            App</title> <style>body {margin-top: 40px; background-color:
            #333;} </style> </head><body> <div
            style=color:white;text-align:center> <h1>Amazon ECS Sample
            App</h1> <h2>Congratulations!</h2> <p>Your application is now
            running on a container in Amazon ECS.</p> </div></body></html>' >
            /usr/local/apache2/htdocs/index.html && httpd-foreground"
        EntryPoint:
          - sh
          - '-c'
        Essential: true
        Image: 'httpd:2.4'
        LogConfiguration:
          LogDriver: awslogs
          Options:
            awslogs-group: /ecs/fargate-task-definition
            awslogs-region: us-east-1
            awslogs-stream-prefix: ecs
        Name: sample-fargate-app
        PortMappings:
          - ContainerPort: 80
            HostPort: 80
            Protocol: tcp
    Cpu: 256
    ExecutionRoleArn: 'arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole'
    Family: task-definition-cfn
    Memory: 512
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX
```

Cluster

Sie können die folgende Vorlage verwenden, um einen leeren Cluster zu erstellen.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
```

```
            "Properties": {
                "ClusterName": "MyEmptyCluster"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: MyEmptyCluster
```

Erstellen mehrerer Amazon ECS-Ressourcen in einem Stack

Sie können die folgende Beispielvorlage verwenden, um mehrere Amazon ECS-Ressourcen in einem Stack zu erstellen. Die Vorlage erstellt einen Amazon ECS-Cluster mit dem Namen `CFNCluster`. Der Cluster enthält eine Linux-Fargate-Aufgabendefinition, die einen Webserver einrichtet. Die Vorlage erstellt auch einen Service mit dem Namen `cfn-service`, der die in der Aufgabendefinition definierte Aufgabe startet und verwaltet. Bevor Sie diese Vorlage verwenden, vergewissern Sie sich, dass die IDs des Subnetzes und der Sicherheitsgruppe in der `NetworkConfiguration` des Services alle zur gleichen VPC gehören und dass die Sicherheitsgruppe über die erforderlichen Regeln verfügt. Weitere Informationen zu Sicherheitsgrupperegeln finden Sie unter Sicherheitsgruppenregeln im Amazon-VPC-Benutzerhandbuch.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "CFNCluster"
            }
        },
        "ECSTaskDefinition": {
```

```
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
                            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS
 Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": "httpd:2.4",
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "awslogs-group": "/ecs/fargate-task-definition",
                                "awslogs-region": "us-east-1",
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": 256,
                "ExecutionRoleArn": "arn:aws:iam::aws_account_id::role/
ecsTaskExecutionRole",
                "Family": "task-definition-cfn",
                "Memory": 512,
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
```

```
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "Properties": {
                "ServiceName": "cfn-service",
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DesiredCount": 1,
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": [
                            "sg-abcdef01234567890"
                        ],
                        "Subnets": [
                            "subnet-abcdef01234567890"
                        ]
                    }
                },
                "TaskDefinition": {
                    "Ref": "ECSTaskDefinition"
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: CFNCluster
  ECSTaskDefinition:
```

```
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: 'httpd:2.4'
          LogConfiguration:
            LogDriver: awslogs
            Options:
              awslogs-group: /ecs/fargate-task-definition
              awslogs-region: us-east-1
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: 256
      ExecutionRoleArn: 'arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole'
      Family: task-definition-cfn
      Memory: 512
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: 'AWS::ECS::Service'
    Properties:
      ServiceName: cfn-service
      Cluster: !Ref ECSCluster
      DesiredCount: 1
      LaunchType: FARGATE
```

```
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - sg-abcdef01234567890
          Subnets:
            - subnet-abcdef01234567890
      TaskDefinition: !Ref ECSTaskDefinition
```

Verwenden von AWS CLI , um Ressourcen aus Vorlagen zu erstellen

Der folgende Befehl erstellt einen Stapel mit dem Namen `ecs-stack` und verwendet eine Vorlagentextdatei mit dem Namen `ecs-template-body.json`. Stellen Sie sicher, dass die Vorlagentextdatei im JSON- oder YAML-Format vorliegt. Der Speicherort der Datei ist im `--template-body`-Parameter. In diesem Fall befindet sich die Vorlagen-Textdatei im aktuellen Verzeichnis.

```
aws cloudformation create-stack \
      --stack-name ecs-stack \
      --template-body file://ecs-template-body.json
```

Um sicherzustellen, dass die Ressourcen korrekt erstellt werden, überprüfen Sie die Amazon ECS-Konsole oder verwenden Sie alternativ die folgenden Befehle:

- Mit dem folgenden Befehl werden alle Aufgabendefinitionen aufgelistet.

  ```
  aws ecs list-task-definitions
  ```

- Mit dem folgenden Befehl werden alle Cluster aufgelistet.

  ```
  aws ecs list-clusters
  ```

- Mit dem folgenden Befehl werden alle Dienste aufgelistet, die im Cluster *CFNCluster*definiert sind. Ersetzen Sie *CFNCluster* mit dem Namen des Clusters, in dem Sie den Service erstellen möchten.

  ```
  aws ecs list-services \
        --cluster CFNCluster
  ```

Erfahren Sie mehr über AWS CloudFormation

Weitere Informationen AWS CloudFormation finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS CloudFormation Benutzerhandbuch
- AWS CloudFormation Benutzerhandbuch für die Befehlszeilenschnittstelle

Erste Schritte mit der Amazon ECS-Befehlszeilenschnittstelle

Amazon ECS hat AWS Copilot veröffentlicht, ein Befehlszeilenschnittstellentool (CLI), das die Erstellung, Veröffentlichung und den Betrieb produktionsreifer containerisierter Anwendungen auf Amazon ECS von einer lokalen Entwicklungsumgebung aus vereinfacht. Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.

Die Amazon Elastic Container Service (Amazon ECS)-Befehlszeilenschnittstelle (CLI) bietet allgemeine Befehle zum einfacheren Erstellen, Aktualisieren und Überwachen von Clustern und Aufgaben von einer lokalen Entwicklungsumgebung aus. Die Amazon ECS-CLI unterstützt Docker-Compose-Dateien, eine beliebte Open-Source-Spezifikation zur Definition und Ausführung von Multi-Container-Anwendungen. Verwenden Sie die ECS-CLI als Teil Ihres täglichen Entwicklungs- und Testzyklus als Alternative zur AWS Management Console.

Die neueste Version der Amazon-ECS-Befehlszeilenschnittstelle unterstützt nur die Hauptversionen der Docker-Compose-Dateisyntax-Versionen 1, 2 und 3. Die in der Compose-Datei angegebene Version muss der Zeichenfolge `"1"`, `"1.0"`, `"2"`, `"2.0"`, `"3"` oder `"3.0"` entsprechen. Docker Compose-Nebenversionen werden nicht unterstützt.

Der Quellcode für die Amazon ECS-CLI ist verfügbar unter GitHub. Dieses Tool wird nicht mehr aktiv weiterentwickelt.

Installieren der Amazon ECS-CLI

Amazon ECS hat AWS Copilot veröffentlicht, ein Befehlszeilenschnittstellentool (CLI), das die Erstellung, Veröffentlichung und den Betrieb produktionsreifer containerisierter Anwendungen auf

Amazon ECS von einer lokalen Entwicklungsumgebung aus vereinfacht. Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.

Die folgenden Schritte zeigen, wie Sie die Amazon-ECS-CLI auf Ihrem macOS-, Linux- oder Windows-System installieren.

So installieren Sie die Amazon-ECS-CLI

1. Laden Sie die Amazon ECS-CLI-Binärdatei herunter.

   macOS

   ```
   sudo curl -Lo /usr/local/bin/ecs-cli https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-darwin-amd64-latest
   ```

   Linux

   ```
   sudo curl -Lo /usr/local/bin/ecs-cli https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-linux-amd64-latest
   ```

   Windows

   Öffnen Sie Windows PowerShell und geben Sie die folgenden Befehle ein.

   > **Note**
   >
   > Wenn Sie auf Berechtigungsprobleme stoßen, stellen Sie sicher, dass Sie Administratorzugriff auf Windows haben und dass Sie PowerShell als Administrator arbeiten.

   ```
   New-Item -Path 'C:\Program Files\Amazon\ECSCLI' -ItemType Directory
   Invoke-WebRequest -OutFile 'C:\Program Files\Amazon\ECSCLI\ecs-cli.exe' https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-windows-amd64-latest.exe
   ```

2. Überprüfen Sie die Amazon-ECS-CLI mithilfe von PGP-Signaturen. Die ausführbaren Amazon ECS-CLI-Dateien werden mithilfe von PGP-Signaturen kryptografisch signiert. Die PGP-

Signaturen können verwendet werden, um die Gültigkeit der ausführbaren Datei von Amazon ECS-CLI zu überprüfen. Verwenden Sie die folgenden Schritte, um die Signaturen mithilfe des GnuPG-Tools zu überprüfen.

a. Laden Sie GnuPG herunter und installieren Sie es. Weitere Informationen finden Sie auf der GnuPG Website.

   macOS

   Wir empfehlen den Einsatz von Homebrew. Installieren Sie Homebrew mithilfe der Anweisungen von ihrer Website. Weitere Informationen finden Sie unter Homebrew. Nachdem Homebrew installiert ist, verwenden Sie den folgenden Befehl aus Ihrem macOS-Terminal:

   ```
   brew install gnupg
   ```

   Linux

   Installieren Sie gpg mit dem Paket-Manager auf Ihrer Linux-Variante.

   Windows

   Laden Sie das einfache Windows-Installationsprogramm von der GnuPG-Website herunter und installieren Sie es als Administrator. Nachdem Sie GnuPG installiert haben, schließen Sie den Administrator und öffnen Sie ihn erneut. PowerShell

   Weitere Informationen finden Sie unter GnuPG-Download.

b. Stellen Sie sicher, dass der GnuPG-Pfad Ihrem Umgebungspfad hinzugefügt wurde.

   macOS

   ```
   echo $PATH
   ```

   Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

   ```
   PATH=$PATH:<path to GnuPG executable files>
   ```

Linux

```
echo $PATH
```

Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Wenn der GnuPG-Pfad in der Ausgabe nicht angezeigt wird, führen Sie den folgenden Befehl aus, um ihn dem Pfad hinzuzufügen.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

c. Erstellen Sie eine lokale Nur-Text-Datei.

macOS

Geben Sie im Terminal Folgendes ein:

```
touch <public_key_filename.txt>
```

Öffnen Sie die Datei mit. TextEdit

Linux

Erstellen Sie eine Textdatei in einem Texteditor wie gedit. Speichern als `public_key_filename.txt`

Windows

Erstellen Sie eine Textdatei in einem Texteditor wie Notepad. Speichern als `public_key_filename.txt`

d. Fügen Sie den folgenden Inhalt des öffentlichen Amazon-ECS-PGP-Schlüssels hinzu und speichern Sie die Datei.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
```

```
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
```

```
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```

Die Details des öffentlichen Amazon ECS-PGP-Schlüssels als Referenz:

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

e. Importieren Sie die Datei mit dem den öffentlichen Amazon-ECS-PGP-Schlüssel mit dem folgenden Befehl im Terminal.

```
gpg --import <public_key_filename.txt>
```

f. Laden Sie die Amazon ECS-CLI-Signaturen herunter. Die Signaturen sind ASCII-getrennte PGP-Signaturen, die in Dateien mit der Erweiterung `.asc` gespeichert sind. Die Signaturen-Datei hat denselben Namen wie die entsprechende ausführbare Datei, mit dem Zusatz `.asc`.

macOS

```
curl -Lo ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-darwin-amd64-latest.asc
```

Linux

```
curl -Lo ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-linux-amd64-latest.asc
```

Windows

```
Invoke-WebRequest -OutFile ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-windows-amd64-latest.exe.asc
```

g. Überprüfen Sie die Signatur.

macOS and Linux

```
gpg --verify ecs-cli.asc /usr/local/bin/ecs-cli
```

Windows

```
gpg --verify ecs-cli.asc 'C:\Program Files\Amazon\ECSCLI\ecs-cli.exe'
```

Erwartete Ausgabe:

```
gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
gpg:                using RSA key DE3CBD61ADAF8B8E
gpg: Good signature from "Amazon ECS <ecs-security@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: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
```

> **Important**
>
> Die Warnung in der Ausgabe wird erwartet und ist unproblematisch. Sie tritt auf, weil keine Vertrauenskette zwischen Ihrem persönlichen PGP-Schlüssel (falls Sie einen haben) und dem Amazon ECS-PGP-Schlüssel vorhanden ist. Weitere Informationen finden Sie unter Web of trust (Netz des Vertrauens).

3. Wenden Sie Ausführungsberechtigungen auf die Binärdatei an.

   macOS and Linux

   ```
   sudo chmod +x /usr/local/bin/ecs-cli
   ```

   Windows

   Bearbeiten Sie die Umgebungsvariablen und fügen Sie `C:\Program Files\Amazon\ECSCLI` in das Variablenfeld `PATH` ein, durch ein Semikolon von vorhandenen Einträgen getrennt. Beispielsweise:

   ```
   setx path "%path%;C:\Program Files\Amazon\ECSCLI"
   ```

   Starten Sie neu, PowerShell damit die Änderungen wirksam werden.

> **Note**
>
> Nachdem die PATH Variable festgelegt wurde, kann die Amazon ECS-CLI entweder von Windows PowerShell oder von der Befehlszeile aus verwendet werden.

4. Überprüfen Sie, ob die CLI korrekt ausgeführt wird.

```
ecs-cli --version
```

Fahren Sie mit Konfigurieren der Amazon ECS-CLI fort.

> **Important**
>
> Sie müssen die Amazon ECS-CLI mit Ihren AWS Anmeldeinformationen, einer AWS Region und einem Amazon ECS-Clusternamen konfigurieren, bevor Sie sie verwenden können. Weitere Informationen finden Sie unter Konfigurieren der Amazon ECS-CLI.

Konfigurieren der Amazon ECS-CLI

Amazon ECS hat AWS Copilot veröffentlicht, ein Befehlszeilenschnittstellentool (CLI), das die Erstellung, Veröffentlichung und den Betrieb produktionsreifer containerisierter Anwendungen auf Amazon ECS von einer lokalen Entwicklungsumgebung aus vereinfacht. Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.

Die Amazon ECS-CLI benötigt einige grundlegende Konfigurationsinformationen, bevor Sie sie verwenden können, z. B. Ihre AWS Anmeldeinformationen, die AWS Region, in der Ihr Cluster erstellt werden soll, und den Namen des zu verwendenden Amazon ECS-Clusters. Konfigurationsinformationen sind in dem Verzeichnis `~/.ecs` auf macOS- und Linux-Systemen und `C:\Users\<username>\AppData\local\ecs` auf Windows-Systemen gespeichert.

So konfigurieren Sie die Amazon ECS-CLI

1. Richten Sie mit dem folgenden Befehl ein CLI-Profil ein und ersetzen *profile_name*Sie es durch Ihren gewünschten Profilnamen *$AWS_ACCESS_KEY_ID* und *$AWS_SECRET_ACCESS_KEY* Umgebungsvariablen durch Ihre AWS Anmeldeinformationen.

   ```
   ecs-cli configure profile --profile-name profile_name --access-
   key $AWS_ACCESS_KEY_ID --secret-key $AWS_SECRET_ACCESS_KEY
   ```

2. Vervollständigen Sie die Konfiguration mit dem folgenden Befehl, wobei Sie *launch_type* durch den standardmäßig zu verwendenden Aufgaben-Launchtyp ersetzen, *region_name* durch die gewünschte AWS -Region, *cluster_name* durch den Namen eines vorhandenen Amazon-ECS-Clusters oder eines neuen zu verwendenden Clusters, und *configuration_name* durch den Namen, den Sie für die Konfiguration verwenden wollen.

   ```
   ecs-cli configure --cluster cluster_name --default-launch-type launch_type --
   region region_name --config-name configuration_name
   ```

Verwendung von Profilen

Die Amazon ECS-CLI unterstützt die Konfiguration mehrerer Sätze von AWS Anmeldeinformationen als benannte Profile mithilfe des ecs-cli configure profile Befehls. Ein Standard-Profil kann mit dem ecs-cli configure profile default Befehl erstellt werden. Diese Profile können dann referenziert werden, wenn Sie Amazon ECS-CLI-Befehle ausführen, für die Anmeldeinformationen erforderlich sind. Verwenden Sie dazu das Flag `--ecs-profile`, andernfalls wird das Standardprofil verwendet.

Verwenden von Cluster-Konfigurationen

Eine Cluster-Konfiguration ist eine Gruppe von Feldern, die einen Amazon ECS-Cluster beschreiben, einschließlich des Namen des Clusters und der Region. Eine Standard-Cluster-Konfiguration kann mit dem ecs-cli configure default Befehl erstellt werden. Die Amazon ECS-CLI unterstützt die Konfiguration mehrerer benannter Cluster-Konfigurationen mit der Option `--config-name`.

Verstehen der Rangfolge

Es gibt mehrere Methoden, um die Anmeldeinformationen und die Region in einem Amazon ECS-CLI-Befehl weiterzugeben. Für jeden davon gilt die folgende Rangfolge.

Die Rangfolge ist für Anmeldeinformationen lautet:

1. Amazon ECS-CLI-Profilflags:
   a. Amazon ECS-Profil (`--ecs-profile`)
   b. AWS Profil (`--aws-profile`)
2. Umgebungsvariablen:
   a. `ECS_PROFILE`
   b. `AWS_PROFILE`
   c. `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` und `AWS_SESSION_TOKEN`
3. ECS config – versucht, die Anmeldeinformationen aus dem ECS-Standardprofil abzurufen.
4. AWS Standardprofil — Versucht, Anmeldeinformationen (`aws_access_key_id`,`aws_secret_access_key`) oder `assume_role` (`role_arn`,`source_profile`) aus dem AWS Profilnamen zu verwenden.
   a. `AWS_DEFAULT_PROFILE` Umgebungsvariable (standardmäßig `default`).
5. EC2-Instance-Rolle

Die Rangfolge für die Region ist:

1. Amazon ECS-CLI-Flags:
   a. Regionflag (`--region`)
   b. Cluster-Konfigurationsflag (`--cluster-config`)
2. ECS-Config – versucht, die Region aus dem ECS-Standardprofil abzurufen.
3. Umgebungsvariablen – versucht, die Region aus den folgenden Umgebungsvariablen abzurufen:
   a. `AWS_REGION`
   b. `AWS_DEFAULT_REGION`
4. AWS profile - versucht, die Region aus dem AWS Profilnamen zu verwenden:
   a. `AWS_PROFILE` Umgebungsvariable
   b. `AWS_DEFAULT_PROFILE` Umgebungsvariable (standardmäßig `default`)

AWS Fargatefür Amazon ECS

AWS Fargate ist eine Technologie, mit der Sie in Amazon ECS Container ausführen können, ohne hierfür Server oder Amazon-EC2-Instance-Cluster verwalten zu müssen. Mit AWS Fargate müssen Sie keine Cluster virtueller Maschinen mehr bereitstellen, konfigurieren oder skalieren, um Container auszuführen. Auf diese Weise müssen keine Servertypen mehr ausgewählt werden, es muss nicht entschieden werden, wann die Cluster skaliert werden oder das Cluster-Packing optimiert werden.

Wenn Sie Ihre Aufgaben und Services mit dem Fargate-Starttyp ausführen, packen Sie Ihre Anwendung in Container, legen die CPU- und Arbeitsspeicheranforderungen fest, definieren Netzwerk- und IAM-Richtlinien und starten die Anwendung. Jede Fargate-Aufgabe hat ihre eigene Isolationsgrenze und teilt den zugrunde liegenden Kernel, die CPU-Ressourcen, die Arbeitsspeicherressourcen oder die Elastic-Network-Schnittstelle nicht mit einer anderen Aufgabe. Sie konfigurieren Ihre Aufgabendefinitionen für Fargate, indem Sie den `requiresCompatibilities`-Aufgabendefinitionsparameter auf `FARGATE` setzen. Weitere Informationen finden Sie unter Starttypen.

Fargate bietet Plattformversionen für Amazon Linux 2 und Microsoft Windows 2019 Server Full- und Core-Editionen an. Sofern nicht anders angegeben, gelten die Informationen auf dieser Seite für alle Fargate-Plattformen.

In diesem Thema werden die verschiedenen Komponenten der Fargate-Aufgaben und -Services beschrieben. Außerdem wird darauf eingegangen, was beim Verwenden von Fargate mit Amazon ECS zu beachten ist.

Weitere Informationen über Regionen, die Linux-Container auf Fargate unterstützen, finden Sie unter the section called “Linux-Container auf AWS Fargate”.

Weitere Informationen zu den Regionen, die Windows-Container auf Fargate unterstützen, finden Sie unter the section called “Windows-Container auf AWS Fargate”.

Anleitungen

Informationen zu den ersten Schritten mit der Konsole finden Sie unter:

- Erfahren Sie, wie Sie eine Amazon ECS-Linux-Aufgabe für den Starttyp Fargate erstellen
- Erfahren Sie, wie Sie eine Amazon ECS-Windows-Aufgabe für den Starttyp Fargate erstellen

Informationen zu den ersten Schritten mit der AWS CLI finden Sie unter:

- Erstellen einer Amazon ECS-Linux-Aufgabe für den Fargate-Starttyp mit dem AWS CLI
- Erstellen einer Amazon ECS-Windows-Aufgabe für den Fargate-Starttyp mit dem AWS CLI

Kapazitätsanbieter

Die folgenden Kapazitätsanbieter sind verfügbar:

- Fargate
- Fargate Spot — Führen Sie unterbrechungstolerante Amazon ECS-Aufgaben zu einem im Vergleich zum Preis reduzierten Preis aus. AWS Fargate Fargate Spot führt Aufgaben über freie Rechenkapazität aus. Wenn die Kapazität wieder AWS benötigt wird, werden Ihre Aufgaben mit einer zweiminütigen Warnung unterbrochen. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den Starttyp Fargate .

  Sie können Fargate Spot nur für Linux-Aufgaben verwenden, die die X86-Architektur verwenden.

Aufgabendefinitionen

Aufgaben, die den Fargate-Starttyp verwenden, unterstützen nicht alle verfügbaren Amazon-ECS-Aufgabendefinitionsparameter. Einige Parameter werden generell nicht unterstützt, bei anderen weicht deren Verhalten für Fargate-Aufgaben ab. Weitere Informationen finden Sie unter CPU und Arbeitsspeicher der Aufgabe.

Plattformversionen

AWS Fargate-Plattformversionen werden verwendet, um auf eine bestimmte Laufzeitumgebung für die Fargate-Task-Infrastruktur zu verweisen. Es handelt sich um eine Kombination aus der Kernel-Version und den Container-Laufzeitversionen. Sie wählen eine Plattformversion aus, wenn Sie eine Aufgabe ausführen oder wenn Sie einen Service erstellen, um eine Reihe identischer Aufgaben zu verwalten.

Neue Revisionen von Plattformversionen werden bei Änderungen der Laufzeitumgebung veröffentlicht, z. B. wenn es Kernel- oder Betriebssystem-Updates, neue Features, Fehlerbehebungen oder Sicherheits-Updates gibt. Eine Fargate-Plattformversion wird aktualisiert,

indem eine neue Plattformversionsrevision erstellt wird. Jede Aufgabe wird während ihres Lebenszyklus auf einer Plattformversionsrevision ausgeführt. Wenn Sie die neueste Version der Plattformversionsrevision verwenden möchten, müssen Sie eine neue Aufgabe starten. Eine neue Aufgabe, die auf Fargate ausgeführt wird, läuft immer auf der neuesten Revision einer Plattformversion, wodurch sichergestellt wird, dass Aufgaben immer auf einer sicheren und gepatchten Infrastruktur gestartet werden.

Wenn ein Sicherheitsproblem gefunden wird, das sich auf eine bestehende Plattformversion auswirkt, wird eine neue gepatchte Version der Plattformversion AWS erstellt und Aufgaben, die auf der anfälligen Version ausgeführt werden, zurückgezogen. Möglicherweise erhalten Sie eine Benachrichtigung, dass die Ausmusterung Ihrer Fargate-Aufgaben geplant wurde. Weitere Informationen finden Sie unter AWS Häufig gestellte Fragen zur Fargate-Aufgabenwartung auf Amazon ECS.

Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS und Fargate Windows-Plattformversionen für Amazon ECS.

Service-Load Balancing

Ihr Amazon-ECS-Service auf AWS Fargate kann optional zur Verwendung von Elastic Load Balancing konfiguriert werden, um Datenverkehr gleichmäßig auf die Aufgaben in Ihrem Service zu verteilen.

Amazon-ECS-Services auf AWS Fargate unterstützen die Load Balancer-Typen Application Load Balancer und Network Load Balancer. Application Load Balancer werden zum Weiterleiten von HTTP/HTTPS-Datenverkehr (oder Layer 7) verwendet. Network Load Balancer werden zum Weiterleiten von TCP- oder UDP-Datenverkehr (oder Layer 4) verwendet. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.

Wenn Sie eine Zielgruppe für diese Services erstellen, müssen Sie zudem `ip` als Zieltyp auswählen, und nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.

Die Verwendung eines Network Load Balancers zur Weiterleitung von UDP-Verkehr an Ihre Amazon ECS on AWS Fargate-Aufgaben wird nur bei Verwendung der Plattformversion 1.4 oder höher unterstützt.

Nutzungsmetriken

Sie können CloudWatch Nutzungsmetriken verwenden, um einen Überblick über die Ressourcennutzung Ihres Kontos zu erhalten. Verwenden Sie diese Kennzahlen, um Ihre aktuelle Servicenutzung in CloudWatch Diagrammen und Dashboards zu visualisieren.

AWS Fargate Die Nutzungsmetriken entsprechen den AWS Servicekontingenten. Sie können Alarme konfigurieren, mit denen Sie benachrichtigt werden, wenn sich Ihre Nutzung einem Servicekontingent nähert. Weitere Hinweise zu Servicekontingenten für AWS Fargate finden Sie unter AWS Fargate Servicekontingenten.

Weitere Informationen zu AWS Fargate Nutzungsmetriken finden Sie unter AWS FargateNutzungsmetriken im Amazon Elastic Container Service-Benutzerhandbuch für AWS Fargate.

Überlegungen zur Amazon ECS-Sicherheit bei der Verwendung des Fargate-Starttyps

Wir empfehlen Kunden, die für ihre Aufgaben eine starke Isolierung suchen, Fargate zu verwenden. Fargate führt jede Aufgabe in einer Hardware-Virtualisierungsumgebung aus. Dadurch wird sichergestellt, dass sich diese containerisierten Workloads keine Netzwerkschnittstellen, kurzlebigen Fargate-Speicher, CPU oder Arbeitsspeicher mit anderen Aufgaben teilen. Weitere Informationen finden Sie unter Sicherheitsüberblick von. AWS Fargate

Bewährte Sicherheitsmethoden von Fargate in Amazon ECS

Wir empfehlen, dass Sie die folgenden bewährten Methoden berücksichtigen, wenn Sie AWS Fargate verwenden. Weitere Hinweise finden Sie unter Sicherheitsüberblick von. AWS Fargate

Wird verwendet AWS KMS , um kurzlebigen Speicher für Fargate zu verschlüsseln

Sie sollten Ihren kurzlebigen Speicher mit verschlüsseln lassen. AWS KMS Für Aufgaben, die mit der Plattformversion `1.4.0` oder höher auf Fargate gehostet werden, erhält jede Aufgabe 20 GiB flüchtigen Speicher. Sie können die Gesamtmenge des flüchtigen Speichers bis zu einem Maximum von 200 GiB erhöhen, indem Sie den `ephemeralStorage`-Parameter in Ihrer Aufgabendefinition angeben. Für solche Aufgaben, die am 28. Mai 2020 oder später gestartet wurden, wird der

kurzlebige Speicher mit einem AES-256-Verschlüsselungsalgorithmus unter Verwendung eines von Fargate verwalteten Verschlüsselungsschlüssels verschlüsselt.

Weitere Informationen finden Sie unter Verwendung von Daten-Volumes in Aufgaben.

Beispiel: Starten einer Aufgabe auf der Fargate-Plattformversion 1.4.0 mit kurzlebiger Speicherverschlüsselung

Der folgende Befehl startet eine Aufgabe auf der Fargate-Plattform Version 1.4. Da diese Aufgabe als Teil des Clusters gestartet wird, verwendet sie den flüchtigen Speicher von 20 GiB, der automatisch verschlüsselt wird.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

SYS_PTRACE-Fähigkeit für Kernel-Syscall-Tracing mit Fargate

Die Standardkonfiguration der Linux-Funktionen, die Ihrem Container hinzugefügt oder entfernt werden, wird von Docker bereitgestellt. Weitere Informationen zu den verfügbaren Funktionen finden Sie unter Laufzeitprivileg und Linux-Funktionen in der Docker-Run-Dokumentation.

Aufgaben, die auf Fargate gestartet werden, unterstützen lediglich das Hinzufügen der SYS_PTRACE-Kernelfunktion.

Das folgende Tutorial-Video zeigt, wie Sie diese Funktion im Sysdig Falco-Projekt verwenden können.

ContainersFromTheCouch - Fehlerbehebung bei Ihrer Fargate-Aufgabe mithilfe der SYS_PTRACE-Funktion

Den im vorherigen Video besprochenen Code finden Sie hier. GitHub

Verwenden Sie Amazon GuardDuty mit Fargate Runtime Monitoring

Amazon GuardDuty ist ein Service zur Bedrohungserkennung, der Ihnen hilft, Ihre Konten, Container, Workloads und die Daten in Ihrer AWS Umgebung zu schützen. Mithilfe von Modellen

für maschinelles Lernen (ML) und Funktionen zur Erkennung von Anomalien und Bedrohungen werden GuardDuty kontinuierlich verschiedene Protokollquellen und Laufzeitaktivitäten überwacht, um potenzielle Sicherheitsrisiken und böswillige Aktivitäten in Ihrer Umgebung zu identifizieren und zu priorisieren.

Runtime Monitoring in GuardDuty schützt Workloads, die auf Fargate ausgeführt werden, indem es die AWS Protokoll- und Netzwerkaktivitäten kontinuierlich überwacht, um bösartiges oder nicht autorisiertes Verhalten zu identifizieren. Runtime Monitoring verwendet einen schlanken, vollständig verwalteten GuardDuty Security Agent, der das Verhalten auf dem Host analysiert, z. B. den Dateizugriff, die Prozessausführung und Netzwerkverbindungen. Dies deckt Probleme wie die Eskalation von Rechten, die Verwendung offengelegter Anmeldeinformationen oder die Kommunikation mit bösartigen IP-Adressen, Domänen und das Vorhandensein von Malware auf Ihren Amazon EC2 EC2-Instances und Container-Workloads ab. Weitere Informationen finden Sie unter GuardDuty Runtime Monitoring im GuardDuty Benutzerhandbuch.

Sicherheitsüberlegungen von Fargate für Amazon ECS

Jede Aufgabe hat eine eigene Infrastrukturkapazität, da Fargate jeden Workload in einer isolierten virtuellen Umgebung ausführt. Workloads, die auf Fargate ausgeführt werden, teilen sich keine Netzwerkschnittstellen, flüchtigen Speicher, CPU oder Arbeitsspeicher mit anderen Aufgaben. Sie können mehrere Container innerhalb einer Aufgabe ausführen, darunter Anwendungs-Container und Beiwagen-Container oder einfach Beiwagen. Ein Beiwagen ist ein Container, der zusammen mit einem Anwendungs-Container in einer Amazon-ECS-Aufgabe ausgeführt wird. Während der Anwendungs-Container den Kernanwendungscode ausführt, können Prozesse, die in Beiwagen ausgeführt werden, die Anwendung erweitern. Mithilfe von Beiwagen können Sie Anwendungsfunktionen in spezielle Container unterteilen, sodass Sie Teile Ihrer Anwendung einfacher aktualisieren können.

Container, die Teil derselben Aufgabe sind, teilen sich Ressourcen für den Fargate-Starttyp, da diese Container immer auf demselben Host laufen und Rechenressourcen gemeinsam nutzen. Diese Container teilen sich auch den von Fargate bereitgestellten flüchtigen Speicher. Linux-Container in einer Aufgabe teilen sich Netzwerk-Namespaces, einschließlich der IP-Adresse und der Netzwerkports. Innerhalb einer Aufgabe können Container, die zur Aufgabe gehören, über localhost miteinander kommunizieren.

Die Laufzeitumgebung in Fargate verhindert, dass Sie bestimmte Controller-Features verwenden, die auf EC2-Instances unterstützt werden. Berücksichtigen Sie Folgendes, wenn Sie Workloads erstellen, die auf Fargate ausgeführt werden:

- Keine privilegierten Container oder privilegierter Zugriff – Features wie privilegierte Container oder privilegierter Zugriff sind derzeit auf Fargate nicht verfügbar. Dies wird sich auf Anwendungsfälle wie die Ausführung von Docker in Docker auswirken.
- Eingeschränkter Zugriff auf Linux-Funktionen – Die Umgebung, in der Container auf Fargate laufen, ist gesperrt. Zusätzliche Linux-Funktionen, wie CAP_SYS_ADMIN und CAP_NET_ADMIN, sind eingeschränkt, um eine Rechteerweiterung zu verhindern. Fargate unterstützt das Hinzufügen der Linux-Funktion CAP_SYS_PTRACE zu Aufgaben, um innerhalb der Aufgabe bereitgestellte Beobachtbarkeits- und Sicherheitstools zur Überwachung der containerisierten Anwendung zu ermöglichen.
- Kein Zugriff auf den zugrundeliegenden Host — Weder Kunden noch AWS Betreiber können eine Verbindung zu einem Host herstellen, auf dem Kunden-Workloads ausgeführt werden. Sie können ECS Exec verwenden, um Befehle in einem auf Fargate laufenden Container auszuführen oder eine Shell für diesen zu erhalten. Sie können ECS Exec verwenden, um Diagnoseinformationen für das Debuggen zu sammeln. Fargate verhindert auch, dass Container auf die Ressourcen des zugrunde liegenden Hosts zugreifen, z. B. auf das Dateisystem, die Geräte, das Netzwerk und die Container-Laufzeit.
- Netzwerke – Sie können Sicherheitsgruppen und Netzwerk-ACLs verwenden, um den ein- und ausgehenden Datenverkehr zu steuern. Fargate-Aufgaben erhalten eine IP-Adresse aus dem konfigurierten Subnetz in Ihrer VPC.

Fargate Linux-Plattformversionen für Amazon ECS

AWS Fargate-Plattformversionen werden verwendet, um auf eine bestimmte Laufzeitumgebung für die Fargate-Task-Infrastruktur zu verweisen. Es handelt sich um eine Kombination aus der Kernel-Version und den Container-Laufzeitversionen. Sie wählen eine Plattformversion aus, wenn Sie eine Aufgabe ausführen oder wenn Sie einen Service erstellen, um eine Reihe identischer Aufgaben zu verwalten.

Neue Revisionen von Plattformversionen werden bei Änderungen der Laufzeitumgebung veröffentlicht, z. B. wenn es Kernel- oder Betriebssystem-Updates, neue Features, Fehlerbehebungen oder Sicherheits-Updates gibt. Eine Fargate-Plattformversion wird aktualisiert, indem eine neue Plattformversionsrevision erstellt wird. Jede Aufgabe wird während ihres Lebenszyklus auf einer Plattformversionsrevision ausgeführt. Wenn Sie die neueste Version der Plattformversionsrevision verwenden möchten, müssen Sie eine neue Aufgabe starten. Eine neue Aufgabe, die auf Fargate ausgeführt wird, läuft immer auf der neuesten Revision einer

Plattformversion, wodurch sichergestellt wird, dass Aufgaben immer auf einer sicheren und gepatchten Infrastruktur gestartet werden.

Wenn ein Sicherheitsproblem gefunden wird, das sich auf eine bestehende Plattformversion auswirkt, wird eine neue gepatchte Version der Plattformversion AWS erstellt und Aufgaben, die auf der anfälligen Version ausgeführt werden, zurückgezogen. Möglicherweise erhalten Sie eine Benachrichtigung, dass die Ausmusterung Ihrer Fargate-Aufgaben geplant wurde. Weitere Informationen finden Sie unter AWS Häufig gestellte Fragen zur Fargate-Aufgabenwartung auf Amazon ECS.

Überlegungen

Bei der Angabe einer Plattformversion sollte Folgendes berücksichtigt werden:

- Bei der Angabe einer Plattformversion können Sie entweder eine bestimmte Versionsnummer verwenden, z. B. `1.4.0`, oder `LATEST`.

  Wenn die LATEST (NEUESTE) Plattformversion ausgewählt ist, wird die Plattformversion `1.4.0` verwendet.
- Wenn Sie die Plattformversion für einen Service aktualisieren möchten, erstellen Sie eine Bereitstellung. Angenommen, Sie haben einen Service, der Aufgaben auf der Linux-Plattformversion `1.3.0` ausführt. Um den Service so zu ändern, dass Aufgaben auf der Linux-Plattformversion `1.4.0` ausgeführt werden, können Sie Ihren Service aktualisieren und eine neue Plattformversion angeben. Ihre Aufgaben werden mit der neuesten Plattformversion und der neuesten Plattformversionsrevision erneut bereitgestellt. Weitere Informationen über Bereitstellungen finden Sie unter Amazon-ECS-Dienstleistungen.
- Wenn Ihr Service skaliert wird, ohne dass die Plattformversion aktualisiert wird, erhalten diese Aufgaben die Plattformversion, die in der aktuellen Bereitstellung des Service angegeben wurde. Angenommen, Sie haben einen Service, der Aufgaben auf der Linux-Plattformversion `1.3.0` ausführt. Wenn Sie die gewünschte Anzahl des Service erhöhen, startet der Service-Scheduler die neuen Aufgaben unter Verwendung der neuesten Plattformversionsrevision von Plattformversion `1.3.0`.
- Neue Aufgaben werden immer auf der neuesten Version einer Plattformversion ausgeführt, wodurch sichergestellt wird, dass Aufgaben immer auf einer gesicherten und gepatchten Infrastruktur gestartet werden.
- Die Plattformversionsnummern für Linux-Container und Windows-Container auf Fargate sind unabhängig voneinander. Beispielsweise sind das Verhalten, die Features und die Software, die in

der Plattformversion `1.0.0` für Windows-Container auf Fargate verwendet werden, nicht mit denen der Plattformversion `1.0.0` für Linux-Container auf Fargate vergleichbar.

Die folgenden sind verfügbare Linux-Plattformversionen. Weitere Informationen zu Plattform-Veraltung finden Sie unter AWS Veraltete Version der Fargate-Linux-Plattform.

1.4.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.4.0`.

- Ab dem 5. November 2020 wird jede neue Amazon-ECS-Aufgabe auf Fargate mithilfe der Plattformversion `1.4.0` gestartet und kann die folgenden Features verwenden:
  - Wenn Sie Secrets Manager zum Speichern von sensiblen Daten verwenden, können Sie einen bestimmten JSON-Schlüssel oder eine bestimmte Version eines Secrets als Umgebungsvariable oder in eine Protokollkonfiguration einfügen. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.
  - Geben Sie Umgebungsvariablen im Massenformat an, indem Sie `environmentFiles`-Containerdefinition-Parameter nutzen. Weitere Informationen finden Sie unter Übergeben Sie eine einzelne Umgebungsvariable an einen Amazon ECS-Container.
  - Aufgaben, die in einer VPC und einem für IPv6 aktiviertem Subnetz ausgeführt werden, werden sowohl eine private IPv4-Adresse als auch eine IPv6-Adresse zugewiesen. Weitere Informationen finden Sie unter Fargate-Aufgabenvernetzung im Benutzerhandbuch zum Amazon Elastic Container Service für AWS Fargate.
  - Der Endpunkt der Aufgabenmetadaten Version 4 stellt zusätzliche Metadaten zu Ihrer Aufgabe und Ihrem Container bereit, einschließlich des Task-Starttyps, des Amazon-Ressourcennamens (ARN) des Containers sowie der verwendeten Protokolltreiber- und Protokolltreiberoptionen. Beim Abfragen des `/stats`-Endpunktes erhalten Sie auch Statistiken zur Netzwerkrate für Ihre Container. Weitere Informationen finden Sie unter Version 4 des Aufgabenmetadaten-Endpunkts.
- Ab dem 30. Juli 2020 wird jede neue Amazon-ECS-Aufgabe, die auf Fargate mithilfe der Plattformversion `1.4.0` gestartet wird, UDP-Datenverkehr mit einem Network Load Balancer an ihre Amazon ECS on Fargate Aufgaben weiterleiten können. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.
- Ab dem 28. Mai 2020 wird bei jeder neuen Amazon ECS-Aufgabe, die mithilfe der Plattformversion auf Fargate gestartet `1.4.0` wird, der kurzlebige Speicher mit einem AES-256-Verschlüsselungsalgorithmus unter Verwendung eines eigenen Verschlüsselungsschlüssels

verschlüsselt. AWS Weitere Informationen finden Sie unter Flüchtiger Speicher für Fargate-Aufgaben für Amazon ECS und Speicheroptionen für Amazon ECS-Aufgaben.

- Unterstützung für die Verwendung von Amazon EFS Dateisystem-Volumes für die persistente Aufgabenspeicherung hinzugefügt. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.
- Der flüchtige Aufgabenspeicher wurde für jede Aufgabe auf mindestens 20 GB erhöht. Weitere Informationen finden Sie unter Flüchtiger Speicher für Fargate-Aufgaben für Amazon ECS.
- Das Verhalten des zu Aufgaben hinführenden und von Aufgaben wegführenden Netzwerkdatenverkehrs wurde aktualisiert. Ab Plattformversion 1.4.0 erhalten alle Fargate-Aufgaben eine einzige Elastic-Network-Schnittstelle (als Aufgaben-ENI bezeichnet) und der gesamte Netzwerkverkehr fließt innerhalb Ihrer VPC durch diese ENI und wird durch Ihre VPC-Flow-Protokolle für Sie sichtbar. Weitere Informationen über Netzwerke für den Amazon EC2 EC2-Starttyp finden Sie unter Fargate Task Networking. Weitere Informationen über Netzwerke für den Fargate-Starttyp finden Sie unterNetzwerkoptionen für Amazon ECS-Aufgaben für den Starttyp Fargate.
- Aufgaben-ENIs fügen Unterstützung für Jumbo-Frames hinzu. Netzwerkschnittstellen sind mit einer Maximum Transmission Unit (MTU) konfiguriert, die der Größe der größten Nutzlast entspricht, die in einen einzelnen Frame passt. Je größer die MTU ist, desto mehr Anwendungsnutzlast passt in ein einzelnes Frame, was den Overhead pro Frame reduziert und die Effizienz erhöht. Die Unterstützung von Jumbo-Frames reduziert den Overhead, wenn der Netzwerkpfad zwischen Ihrer Aufgabe und dem Ziel Jumbo-Frames unterstützt, wie z. B. den gesamten Datenverkehr, der in Ihrer VPC verbleibt.
- CloudWatch Container Insights wird Netzwerkleistungskennzahlen für Fargate-Aufgaben enthalten. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.
- Unterstützung für den Aufgabenmetadaten-Endpunkt Version 4 hinzugefügt, der zusätzliche Informationen für Ihre Fargate-Aufgaben enthält, einschließlich Netzwerkstatistiken und in welcher Availability Zone die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter > Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4 undAmazon ECS-Endpunkt für Aufgabenmetadaten, Version 4, für Aufgaben auf Fargate.
- Unterstützung für den Linux-Parameter`SYS_PTRACE` in Containerdefinitionen hinzugefügt. Weitere Informationen finden Sie unter Linux-Parameter.
- Der Fargate Container-Agent ersetzt die Verwendung des Amazon-ECS-Container-Agents für alle Fargate-Aufgaben. Diese Änderung hat normalerweise keine Auswirkungen auf die Ausführung Ihrer Aufgaben.

- Die Containerlaufzeit verwendet nun Containerd anstelle von Docker. Diese Änderung hat wahrscheinlich keine Auswirkungen auf die Ausführung Ihrer Aufgaben. Sie werden feststellen, dass einige Fehlermeldungen, die in der Containerlaufzeit entstehen, jetzt nicht mehr Docker, sondern allgemeinere Fehlern erwähnen. Weitere Informationen finden Sie unter Gestoppte Aufgaben-Fehlercodes im Amazon Elastic Container Service Benutzerhandbuch für AWS Fargate.
- Basierend auf Amazon Linux 2.

1.3.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.3.0`.

- Ab dem 30. September 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, den Protokolltreiber `awsfirelens`. Konfigurieren Sie, dass Amazon ECS die FireLens Aufgabendefinitionsparameter verwendet, um Protokolle zur Protokollspeicherung und Analyse an ein AWS Service- oder AWS Partnernetzwerkziel (APN) weiterzuleiten. Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.
- Aufgabenrecycling für Fargate-Aufgaben hinzugefügt. Dies ist der Prozess der Aktualisierung von Aufgaben, die Teil eines Amazon-ECS-Service sind. Weitere Informationen finden Sie unter Aufgaben-Wartung im Amazon Elastic Container Service-Benutzerhandbuch für AWS Fargate.
- Ab dem 27. März 2019 bietet jede neue gestartete Fargate-Aufgabe zusätzliche Aufgabendefinitions-Parameter, mit denen Sie eine Proxy-Konfiguration, Abhängigkeiten für das Startup und Herunterfahren von Containern sowie einen Zeitbeschränkungs-Wert für das Starten und Stoppen pro Container definieren können. Weitere Informationen finden Sie unter Proxykonfiguration, Container-Abhängigkeit und Container-Timeouts.
- Ab dem 2. April 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, das Injizieren sensibler Daten in Ihre Container, indem Ihre sensiblen Daten entweder in AWS Secrets Manager Secrets- oder AWS Systems Manager Parameter Store-Parametern gespeichert und dann in Ihrer Container-Definition referenziert werden. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.
- Ab dem 1. Mai 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, das Verweisen auf sensible Daten in der Protokollkonfiguration eines Containers mithilfe der `secretOptions`-Containerdefinitionsparameter. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

- Ab dem 1. Mai 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, `densplunk` - Protokolltreiber zusätzlich zum `awslogs`-Protokolltreiber. Weitere Informationen finden Sie unter Speicher und Protokollierung.
- Ab dem 9. Juli 2019 unterstützen alle neuen Fargate-Aufgaben, die gestartet werden, CloudWatch Container Insights. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.
- Ab dem 3. Dezember 2019 wird der Spot-Kapazitätsanbieter Fargate Spot unterstützt. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den Starttyp Fargate .
- Basierend auf Amazon Linux 2.

Migration zur Linux-Plattformversion 1.4.0

Folgendes sollte berücksichtigt werden, wenn Sie Ihre Amazon-ECS-Aufgaben auf Fargate von Plattformversion `1.0.0`, `1.1.0`, `1.2.0` oder `1.3.0` zur Plattformversion `1.4.0` migrieren: Es wird als Best Practice angesehen, um sicherzustellen, dass Ihre Aufgabe auf der Plattformversion `1.4.0` ordnungsgemäß funktioniert, bevor Sie Ihre Aufgaben migrieren.

- Das Verhalten des zu Aufgaben hinführenden und von Aufgaben wegführenden Netzwerkdatenverkehrs wurde aktualisiert. Ab Plattformversion 1.4.0 erhalten alle Amazon ECS auf Fargate-Aufgaben eine einzige Elastic-Network-Schnittstelle (als Aufgaben-ENI bezeichnet) und der gesamte Netzwerkverkehr fließt innerhalb Ihrer VPC durch diese ENI und wird durch Ihre VPC-Flow-Protokolle für Sie sichtbar. Weitere Informationen finden Sie unter Netzwerkoptionen für Amazon ECS-Aufgaben für den Starttyp Fargate.
- Wenn Sie Schnittstellen-VPC-Endpunkte verwenden, sollten Sie Folgendes beachten:
  - Bei Verwendung von Container-Images, die mit Amazon ECR gehostet werden, sind sowohl die com.amazonaws.region.ecr.dkr und com.amazonaws.region.ecr.api Amazon ECR-VPC Endpunkte sowie der Amazon S3 Gateway-Endpunkt erforderlich. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Wenn Sie eine Aufgabendefinition verwenden, die auf Secrets Manager-Secrets verweist, um vertrauliche Daten für Ihre Container abzurufen, müssen Sie die Schnittstellen-VPC-Endpunkte für Secrets Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Secrets Manager mit VPC-Endpunkten im AWS Secrets Manager -Benutzerhandbuch.
  - Wenn Sie eine Aufgabendefinition verwenden, die Parameter des Systems Manager-Parametespeichers referenziert, um vertrauliche Daten für Ihre Container abzurufen, müssen Sie

die Schnittstellen-VPC-Endpunkte für Systems Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Systems Manager mit VPC-Endpunkten im AWS Systems Manager - Benutzerhandbuch.

- Stellen Sie sicher, dass die Sicherheitsgruppe in der Elastic Network Interface (ENI), die Ihrer Aufgabe zugeordnet ist, über Sicherheitsgruppenregeln verfügt, die den Datenverkehr zwischen der Aufgabe und den von Ihnen verwendeten VPC-Endpunkten zulassen.

AWS Veraltete Version der Fargate-Linux-Plattform

Diese Seite listet Linux-Plattformversionen auf, die AWS Fargate als veraltet markiert hat oder deren Verfall geplant ist. Diese Plattformversionen bleiben bis zum veröffentlichten Veraltungs-Datum verfügbar.

Ein erzwungenes Aktualisierungsdatum wird für jede Plattformversion bereitgestellt, für die eine Veraltung vorgesehen ist. Am erzwungenen Aktualisierungsdatum wird jeder Dienst, der die `LATEST`-Plattformversion, die auf eine Plattformversion verweist, die für eine Veraltung vorgesehen ist, nutzt, mit der Option „neue Bereitstellung erzwingen“ aktualisiert. Wenn der Service mit der Option „Neue Bereitstellung erzwingen“ aktualisiert wird, werden alle Tasks, die auf einer Plattformversion ausgeführt werden, die für die Veraltung vorgesehen ist, angehalten und neue Tasks werden mit der Plattformversion gestartet, auf die das `LATEST`-Tag zu diesem Zeitpunkt verweist. Eigenständige Aufgaben oder Dienste mit einer expliziten Plattformversion sind vom erzwungenen Aktualisierungsdatum nicht betroffen.

Wir empfehlen, Ihre eigenständigen Servicetasks zu aktualisieren, um die neueste Plattformversion zu verwenden. Weitere Informationen zum Migrieren auf die neueste Plattformversion finden Sie unter Migration zur Linux-Plattformversion 1.4.0.

Sobald eine Plattformversion das Datum der Veraltung erreicht, ist die Plattformversion für neue Aufgaben oder Dienste nicht mehr verfügbar. Alle eigenständigen Aufgaben oder Dienste, die explizit eine veraltete Plattformversion verwenden, verwenden diese Plattformversion weiter, bis die Aufgaben beendet sind. Nach dem Veraltungsdatum erhält eine veraltete Plattformversion keine Sicherheitsupdates oder Fehlerbehebungen mehr.

Plattformversion	Datum der Aktualisierung erzwingen	Datum der Veraltung
1.0.0	26. Oktober 2020	14. Dezember 2020

Plattformversion	Datum der Aktualisierung erzwingen	Datum der Veraltung
1.1.0	26. Oktober 2020	14. Dezember 2020
1.2.0	26. Oktober 2020	14. Dezember 2020

Weitere Informationen zu aktuellen Plattformversionen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

Changelog für veraltete AWS Fargate-Linux-Versionen

1.2.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.2.0`.

> **Note**
>
> Die Plattformversion `1.2.0` steht nicht mehr zur Verfügung. Weitere Informationen zu Plattform-Veraltung finden Sie unter AWS Veraltete Version der Fargate-Linux-Plattform.

- Unterstützung für die Authentifizierung privater Registrierungen mit hinzugefügt. AWS Secrets Manager Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.

1.1.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.1.0`.

> **Note**
>
> Die Plattformversion `1.1.0` steht nicht mehr zur Verfügung. Weitere Informationen zu Plattform-Veraltung finden Sie unter AWS Veraltete Version der Fargate-Linux-Plattform.

- Zusätzliche Unterstützung für den Metadatenendpunkt der Amazon-ECS-Aufgaben. Weitere Informationen finden Sie unter Amazon ECS-Aufgabenmetadaten für Aufgaben auf Fargate verfügbar.

- Unterstützung von Docker-Zustandsprüfungen in Container-Definitionen hinzugefügt. Weitere Informationen finden Sie unter Zustandsprüfung.
- Zusätzliche Unterstützung für Amazon-ECS-Service-Discovery. Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.

1.0.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.0.0`.

> **Note**
>
> Die Plattformversion `1.0.0` steht nicht mehr zur Verfügung. Weitere Informationen zu Plattform-Veraltung finden Sie unter AWS Veraltete Version der Fargate-Linux-Plattform.

- Basierend auf Amazon Linux 2017.09.
- Erstversion.

Pull-Verhalten von Linux-Containern auf Fargate-Container-Images für Amazon ECS

Jede Fargate-Aufgabe wird auf einer eigenen Single-Use-, Single-Tenant-Instance ausgeführt. Wenn Sie Linux-Container auf Fargate ausführen, werden Container-Images oder Container-Image-Ebenen nicht auf der Instance zwischengespeichert. Daher muss für jedes in der Aufgabe definierte Container-Image für jede Fargate-Aufgabe das gesamte Container-Image aus der Container-Image-Registry abgerufen werden. Die Zeit, die zum Abrufen der Bilder benötigt wird, steht in direktem Zusammenhang mit der Zeit, die zum Starten einer Fargate-Aufgabe benötigt wird.

Berücksichtigen Sie Folgendes, um die Abrufzeit von Bildern zu optimieren.

Nähe zum Container-Bild

Um die Zeit zu reduzieren, die für das Herunterladen von Container-Images benötigt wird, sollten Sie die Daten so nah wie möglich am Computer platzieren. Das Abrufen eines Container-Images über das Internet oder über das Internet AWS-Regionen kann sich auf die Download-Zeit auswirken. Es wird empfohlen, das Container-Image in derselben Region zu speichern, in der die

Aufgabe ausgeführt wird. Wenn Sie das Container-Image in Amazon ECR speichern, verwenden Sie einen VPC-Schnittstellenendpunkt, um die Abrufzeit des Images weiter zu reduzieren. Weitere Informationen finden Sie unter VPC-Endpunkte der Amazon ECR-Schnittstelle (AWS PrivateLink) im Amazon ECR-Benutzerhandbuch.

Reduzierung der Größe von Container-Bildern

Die Größe eines Container-Images wirkt sich direkt auf die Download-Zeit aus. Durch die Reduzierung der Größe des Container-Images oder der Anzahl der Container-Image-Ebenen kann die Zeit reduziert werden, die für das Herunterladen eines Images benötigt wird. Lightweight-Basis-Images (wie das minimale Container-Image von Amazon Linux 2023) können erheblich kleiner sein als solche, die auf herkömmlichen Betriebssystem-Basis-Images basieren. Weitere Informationen zum Minimal-Image finden Sie unter AL2023 Minimal Container Image im Amazon Linux 2023 User Guide.

Alternative Komprimierungsalgorithmen

Container-Image-Ebenen werden häufig komprimiert, wenn sie in eine Container-Image-Registry übertragen werden. Durch die Komprimierung der Container-Image-Ebene wird die Datenmenge reduziert, die über das Netzwerk übertragen und in der Container-Image-Registry gespeichert werden muss. Nachdem eine Container-Image-Ebene von der Container-Runtime auf eine Instanz heruntergeladen wurde, wird diese Ebene dekomprimiert. Der verwendete Komprimierungsalgorithmus und die Anzahl der für die Laufzeit verfügbaren vCPUs wirken sich auf die Zeit aus, die zum Dekomprimieren des Container-Images benötigt wird. Auf Fargate können Sie die Größe der Aufgabe erhöhen oder den leistungsfähigeren zstd-Komprimierungsalgorithmus nutzen, um die für die Dekomprimierung benötigte Zeit zu reduzieren. Weitere Informationen finden Sie unter ztsd on. GitHub Informationen zur Implementierung der Images für Fargate finden Sie unter Reducing AWS Fargate Startup Times with zstd Compressed Container Images.

Container-Images werden verzögert geladen

Bei großen Container-Images (> 250 MB) kann es optimal sein, ein Container-Image verzögert zu laden, anstatt das gesamte Container-Image herunterzuladen. Auf Fargate können Sie Seekable OCI (SOCI) verwenden, um ein Container-Image verzögert aus einer Container-Image-Registry zu laden. Weitere Informationen finden Sie unter soci-snapshotter on GitHub und Lazy loading container images using Seekable OCI (SOCI).

Fargate Windows-Plattformversionen für Amazon ECS

AWS Fargate-Plattformversionen werden verwendet, um auf eine bestimmte Laufzeitumgebung für die Fargate-Task-Infrastruktur zu verweisen. Es handelt sich um eine Kombination aus der Kernel-Version und den Container-Laufzeitversionen. Sie wählen eine Plattformversion aus, wenn Sie eine Aufgabe ausführen oder wenn Sie einen Service erstellen, um eine Reihe identischer Aufgaben zu verwalten.

Neue Revisionen von Plattformversionen werden bei Änderungen der Laufzeitumgebung veröffentlicht, z. B. wenn es Kernel- oder Betriebssystem-Updates, neue Features, Fehlerbehebungen oder Sicherheits-Updates gibt. Eine Fargate-Plattformversion wird aktualisiert, indem eine neue Plattformversionsrevision erstellt wird. Jede Aufgabe wird während ihres Lebenszyklus auf einer Plattformversionsrevision ausgeführt. Wenn Sie die neueste Version der Plattformversionsrevision verwenden möchten, müssen Sie eine neue Aufgabe starten. Eine neue Aufgabe, die auf Fargate ausgeführt wird, läuft immer auf der neuesten Revision einer Plattformversion, wodurch sichergestellt wird, dass Aufgaben immer auf einer sicheren und gepatchten Infrastruktur gestartet werden.

Wenn ein Sicherheitsproblem gefunden wird, das sich auf eine bestehende Plattformversion auswirkt, wird eine neue gepatchte Version der Plattformversion AWS erstellt und Aufgaben, die auf der anfälligen Version ausgeführt werden, zurückgezogen. Möglicherweise erhalten Sie eine Benachrichtigung, dass die Ausmusterung Ihrer Fargate-Aufgaben geplant wurde. Weitere Informationen finden Sie unter AWS Häufig gestellte Fragen zur Fargate-Aufgabenwartung auf Amazon ECS.

Hinweise zur Plattformversion

Bei der Angabe einer Plattformversion sollte Folgendes berücksichtigt werden:

- Bei der Angabe einer Plattformversion können Sie entweder eine bestimmte Versionsnummer verwenden, z. B. `1.0.0`, oder `LATEST`.

  Wenn die LATEST (NEUESTE) Plattformversion ausgewählt ist, wird die Plattform `1.0.0` verwendet.
- Neue Aufgaben werden immer auf der neuesten Version einer Plattformversion ausgeführt, wodurch sichergestellt wird, dass Aufgaben immer auf einer gesicherten und gepatchten Infrastruktur gestartet werden.

- Microsoft-Windows-Server-Container-Images müssen von einer bestimmten Version von Windows Server erstellt werden. Sie müssen dieselbe Version von Windows Server in der `platformFamily` auswählen, wenn Sie eine Aufgabe ausführen oder einen Service erstellen, der dem Windows-Server-Container-Image entspricht. Außerdem können Sie in der Aufgabendefinition eine passende `operatingSystemFamily` angeben, um zu verhindern, dass Aufgaben auf der falschen Windows-Version ausgeführt werden. Weitere Informationen finden Sie unter Abgleichen der Container-Host-Version mit Container-Image-Versionen auf der Microsoft-Learn-Website.
- Die Plattformversionsnummern für Linux-Container und Windows-Container auf Fargate sind unabhängig voneinander. Beispielsweise sind das Verhalten, die Features und die Software, die in der Plattformversion `1.0.0` für Windows-Container auf Fargate verwendet werden, nicht mit denen der Plattformversion `1.0.0` für Linux-Container auf Fargate vergleichbar.

Die folgenden sind verfügbare Plattformversionen für Windows-Container.

1.0.0

Im Folgenden finden Sie das Änderungsprotokoll für die Plattformversion `1.0.0`.

- Erstveröffentlichung zur Support auf den folgenden Microsoft-Windows-Server-Betriebssystemen:
  - Windows Server 2019 Voll
  - Windows Server 2019 Kern
  - Windows Server 2022 Voll
  - Windows Server 2022 Kern

Überlegungen zu Windows-Containern auf Fargate für Amazon ECS

Im Folgenden sind die Unterschiede und Überlegungen aufgeführt, die Sie beachten sollten, wenn Sie Windows-Container auf AWS Fargate ausführen.

Wenn Sie Aufgaben auf Linux- und Windows-Containern ausführen müssen, müssen Sie separate Aufgabendefinitionen für jedes Betriebssystem erstellen.

AWS übernimmt die Verwaltung der Betriebssystem-Lizenzen, sodass Sie keine zusätzlichen Microsoft Windows Server-Lizenzen benötigen.

Windows Containers on AWS Fargate unterstützt die folgenden Betriebssysteme:

- Windows Server 2019 Voll
- Windows Server 2019 Kern
- Windows Server 2022 Voll
- Windows Server 2022 Kern

Windows-Container auf AWS Fargate unterstützen den awslogs-Treiber. Weitere Informationen finden Sie unter the section called “Protokolle senden an CloudWatch ”.

Die folgenden Features werden auf Windows-Containern auf Fargate nicht unterstützt:

- Gruppenverwaltete Service-Konten (gMSA)
- Amazon FSx
- ENI Trunking
- App-Mesh-Service und Proxy-Integration für Aufgaben
- Firelens-Protokollrouter-Integration für Aufgaben
- EFS-Volumes
- Die folgenden Aufgabendefinitionsparameter:
  - `maxSwap`
  - `swappiness`
  - `environmentFiles`
- Der Fargate-Spot-Kapazitätsanbieter
- Image-Volumes

  Die Dockerfile-Option `volume` wird ignoriert. Nutzen Sie stattdessen eine Bind-Bindungsbereitstellung in Ihrer Aufgabendefinition. Weitere Informationen finden Sie unter Bind-Mounts mit Amazon ECS verwenden.

Windows-Container auf Fargate-Container-Image-Pull-Verhalten für Amazon ECS

Fargate Windows speichert das von Microsoft bereitgestellte Servercore-Basisimage des letzten Monats und des Vormonats im Cache. Diese Images entsprechen den Patches mit der KB/Build-Nummer, die an jedem Patch-Dienstag aktualisiert werden. Beispielsweise hat Microsoft am

04.09.2024 KB5036896 (17763.5696) für Windows Server 2019 veröffentlicht. Der vorherige Monat KB am 03.12.2024 war KB5035849 (17763.5576). Also wurden für die Plattformen und die folgenden Container-Images zwischengespeichert: `WINDOWS_SERVER_2019_CORE` `WINDOWS_SERVER_2019_FULL`

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5696`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5576`

Darüber hinaus veröffentlichte Microsoft am 04.09.2024 KB5036909 (20348.2402) für Windows Server 2022. In den vergangenen Monaten war KB am 03.12.2024 KB5035857 (20348.2340). Für die Plattformen und die folgenden Container wurden also Images zwischengespeichert: `WINDOWS_SERVER_2022_CORE` `WINDOWS_SERVER_2022_FULL`

- `mcr.microsoft.com/windows/servercore:ltsc2022`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2402`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2340`

Flüchtiger Speicher für Fargate-Aufgaben für Amazon ECS

Bei der Bereitstellung AWS Fargate erhält jede Amazon ECS-Aufgabe, die auf Linux-Containern gehostet wird, den folgenden kurzlebigen Speicher für Bind-Mounts. Diese können aufgespielt und mit den Parametern `volumes`, `mountPoints` und `volumesFrom` in der Aufgabendefinition von allen Containern verwendet werden. Dies wird bei aktivierten Windows-Containern nicht unterstützt. AWS Fargate

Plattformversionen für Fargate-Linux-Container

Version 1.4.0 oder höher

Standardmäßig erhalten Amazon-ECS-Aufgaben, die auf Fargate mit Plattformversion `1.4.0` oder höher gehostet werden mindestens 20 GiB flüchtigen Speicher. Die Gesamtmenge des flüchtigen Speichers kann bis zu einem Maximum von 200 GiB erhöht werden. Dazu können Sie den `ephemeralStorage`-Parameter in Ihrer Aufgabendefinition festlegen.

Das gezogene, komprimierte und das unkomprimierte Container-Image für die Aufgabe wird im flüchtigen Speicher gespeichert. Um die Gesamtmenge des flüchtigen Speichers zu ermitteln, die Ihre

Aufgabe verwenden muss, müssen Sie die Speichermenge, die Ihr Containerimage verwendet, von der Gesamtmenge des flüchtigen Speichers abziehen, der Ihrer Aufgabe zugewiesen ist.

Bei Aufgaben mit Plattformversion `1.4.0` oder höher, die am 28. Mai 2020 oder später gestartet wurden, wird der flüchtige Speicher mit einem AES-256-Verschlüsselungsalgorithmus verschlüsselt. Dieser Algorithmus verwendet einen AWS eigenen Verschlüsselungsschlüssel, oder Sie können Ihren eigenen, vom Kunden verwalteten Schlüssel erstellen. Weitere Informationen finden Sie unter Vom Kunden verwaltete Schlüssel für AWS Fargate kurzlebigen Speicher.

Für Aufgaben, die Plattformversion `1.4.0` oder höher verwenden und die am 18. November 2022 oder später gestartet wurden, wird die flüchtige Speichernutzung über den Aufgabenmetadaten-Endpunkt gemeldet. Ihre Anwendungen in Ihren Aufgaben können den Aufgabenmetadaten-Endpunkt-Version 4 abfragen, um deren reservierte Größe für flüchtigen Speicher und die verwendete Menge abzurufen.

Darüber hinaus werden die Größe des reservierten flüchtigen Speichers und die verwendete Menge an Container Insights gesendet, wenn Sie Amazon CloudWatch Container Insights aktivieren.

> **Note**
>
> Fargate reserviert Speicherplatz auf der Festplatte. Er wird nur von Fargate verwendet. Ihnen entstehen dafür keine Kosten. Es wird in diesen Metriken nicht angezeigt. Sie können diesen zusätzlichen Speicherplatz jedoch in anderen Tools sehen, wie z. B. `df`.

Version 1.3.0 oder früher

Für Amazon-ECS-Aufgaben in Fargate, die Plattformversion `1.3.0` oder früher verwenden, erhält jede Aufgabe den folgenden flüchtigen Speicher.

- 10 GB Speicher auf Docker-Ebene

  > **Note**
  >
  > Dieser Betrag umfasst sowohl komprimierte als auch unkomprimierte Container-Image-Artifakte.

- Zusätzliche 4 GB für Volumen-Mounts. Diese können aufgespielt und mit den Parametern `volumes`, `mountPoints` und `volumesFrom` in der Aufgabendefinition von allen Containern verwendet werden.

Plattformversionen für Fargate-Windows-Container

Version 1.0.0 oder höher

Standardmäßig erhalten Amazon-ECS-Aufgaben, die auf Fargate mit Plattformversion `1.0.0` oder höher gehostet werden mindestens 20 GiB flüchtigen Speicher. Die Gesamtmenge des flüchtigen Speichers kann bis zu einem Maximum von 200 GiB erhöht werden. Dazu können Sie den `ephemeralStorage`-Parameter in Ihrer Aufgabendefinition festlegen.

Das gezogene, komprimierte und das unkomprimierte Container-Image für die Aufgabe wird im flüchtigen Speicher gespeichert. Um die Gesamtmenge des flüchtigen Speichers zu ermitteln, die Ihre Aufgabe verwenden muss, müssen Sie die Speichermenge, die Ihr Containerimage verwendet, von der Gesamtmenge des flüchtigen Speichers abziehen, der Ihrer Aufgabe zugewiesen ist.

Weitere Informationen finden Sie unter Bind-Mounts mit Amazon ECS verwenden.

Vom Kunden verwaltete Schlüssel für AWS Fargate kurzlebigen Speicher

AWS Fargate unterstützt vom Kunden verwaltete Schlüssel zur Verschlüsselung von Daten für Amazon ECS-Aufgaben, die in kurzlebigen Speichern gespeichert sind, um Kunden, die Vorschriften beachten, bei der Einhaltung ihrer internen Sicherheitsrichtlinien zu unterstützen. Kunden profitieren weiterhin von den Vorteilen von Fargate ohne Server und bieten Compliance-Auditoren gleichzeitig mehr Einblick in die selbstverwaltete Speicherverschlüsselung. Fargate verfügt zwar standardmäßig über eine von Fargate verwaltete kurzlebige Speicherverschlüsselung, Kunden können jedoch auch ihre eigenen selbstverwalteten Schlüssel verwenden, um sensible Daten wie Finanz- oder Gesundheitsinformationen zu verschlüsseln.

Sie können Ihre eigenen Schlüssel importieren oder die Schlüssel in erstellen. AWS KMS AWS KMS Diese selbstverwalteten Schlüssel werden in AWS KMS standardmäßigen AWS KMS Lebenszyklusaktionen wie Rotation, Deaktivierung und Löschen gespeichert und führen diese aus. Sie können den Zugriff auf und die Verwendung von Schlüsseln in CloudTrail Protokollen überprüfen.

Standardmäßig unterstützt der KMS-Schlüssel 50.000 Grants pro Schlüssel. Fargate verwendet einen einzigen AWS KMS Zuschuss pro vom Kunden verwalteter Schlüsselaufgabe und unterstützt somit bis zu 50.000 gleichzeitige Aufgaben für einen Schlüssel. Wenn Sie diese Anzahl erhöhen möchten, können Sie eine Erhöhung des Limits beantragen, die auf einer case-by-case bestimmten Grundlage genehmigt wird.

Fargate berechnet keine zusätzlichen Gebühren für die Verwendung von kundenverwalteten Schlüsseln. Ihnen wird nur der Standardpreis für die Verwendung von AWS KMS Schlüsseln für Speicher- und API-Anfragen berechnet.

Themen

Erstellen Sie einen Verschlüsselungsschlüssel für Fargate Ephemeral Storage

> **Note**
>
> Die kurzlebige Speicherverschlüsselung von Fargate mit vom Kunden verwalteten Schlüsseln ist für Windows-Taskcluster nicht verfügbar.
> Die kurzlebige Speicherverschlüsselung von Fargate mit vom Kunden verwalteten Schlüsseln ist frühestens verfügbar`platformVersions`. `1.4.0`
> Fargate reserviert Speicherplatz auf einem kurzlebigen Speicher, der nur von Fargate genutzt wird, und Ihnen wird der Speicherplatz nicht in Rechnung gestellt. Die Zuteilung kann sich von denen unterscheiden, die nicht vom Kunden verwaltet werden, aber der Gesamtspeicher bleibt gleich. Sie können sich diese Änderung in Tools wie `df` anzeigen lassen.

Gehen Sie wie folgt vor, um einen vom Kunden verwalteten Schlüssel (CMK) zur Verschlüsselung von kurzlebigem Speicher für Fargate in AWS KMS zu erstellen.

1. Navigieren Sie zu https://console.aws.amazon.com/kms.
2. Folgen Sie den Anweisungen zum Erstellen von Schlüsseln im AWS Key Management Service Entwicklerhandbuch.
3. Achten Sie bei der Erstellung Ihres AWS KMS Schlüssels darauf, dass Sie in den wichtigsten Richtlinien die für den Fargate-Dienst relevanten AWS KMS Betriebsberechtigungen angeben. Die folgenden API-Operationen müssen in der Richtlinie zulässig sein, um Ihren vom Kunden verwalteten Schlüssel mit Ihren Amazon ECS-Clusterressourcen verwenden zu können.

    - `kms:GenerateDataKeyWithoutPlainText`- Rufen Sie `GenerateDataKeyWithoutPlainText` auf, um aus dem bereitgestellten Schlüssel einen verschlüsselten AWS KMS Datenschlüssel zu generieren.

- `kms:CreateGrant`- Fügt einem vom Kunden verwalteten Schlüssel einen Zuschuss hinzu. Gewährt Kontrollzugriff auf einen bestimmten AWS KMS Schlüssel, der den Zugriff auf Grant-Operationen ermöglicht, die Amazon ECS Fargate benötigt. Weitere Informationen zur Verwendung von Grants finden Sie im AWS Key Management Service Developer Guide. Dadurch kann Amazon ECS Fargate Folgendes tun:
  - Rufen Sie an`Decrypt`, AWS KMS um den Verschlüsselungsschlüssel zum Entschlüsseln der kurzlebigen Speicherdaten zu erhalten.
  - Richten Sie einen Principal ein, der in den Ruhestand geht, damit der Dienst `RetireGrant`
- `kms:DescribeKey`- Stellt dem Kunden verwaltete Schlüsseldetails zur Verfügung, damit Amazon ECS den Schlüssel validieren kann, sofern er symmetrisch und aktiviert ist.

Das folgende Beispiel zeigt eine AWS KMS Schlüsselrichtlinie, die Sie auf den Zielschlüssel für die Verschlüsselung anwenden würden. Um die Beispielrichtlinien zu verwenden, ersetzen Sie die *`Platzhalter für Benutzereingaben`* durch Ihre eigenen Informationen. Konfigurieren Sie wie immer nur die Berechtigungen, die Sie benötigen.

```
{
      "Sid": "Allow generate data key access for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow grant creation permission for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
```

```
      "Action": [
        "kms:CreateGrant"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        },
       "ForAllValues:StringEquals": {
          "kms:GrantOperations": [
             "Decrypt"
          ]
       }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow describe key permission for cluster operator - CreateCluster
 and UpdateCluster.",
      "Effect": "Allow",
      "Principal": { "AWS":"arn:aws:iam::customerAccountId:role/
ClusterOperatorRole" },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
```

Fargate-Aufgaben verwenden die Schlüssel `aws:ecs:clusterAccount` und den `aws:ecs:clusterName` Verschlüsselungskontext für kryptografische Operationen mit dem Schlüssel. Kunden sollten diese Berechtigungen hinzufügen, um den Zugriff auf ein bestimmtes Konto und/oder einen bestimmten Cluster einzuschränken.

Weitere Informationen finden Sie unter Verschlüsselungskontext im AWS KMS - Entwicklerhandbuch.

Beim Erstellen oder Aktualisieren eines Clusters haben Sie die Möglichkeit, den Bedingungsschlüssel zu verwenden`fargateEphemeralStorageKmsKeyId`. Dieser

Bedingungsschlüssel ermöglicht Kunden eine genauere Kontrolle über die IAM-Richtlinien. Aktualisierungen der `fargateEphemeralStorageKmsKeyId` Konfiguration werden nur bei neuen Servicebereitstellungen wirksam.

Im Folgenden finden Sie ein Beispiel dafür, wie Kunden nur einem bestimmten Satz genehmigter AWS KMS Schlüssel Berechtigungen gewähren können.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ecs:fargate-ephemeral-storage-kms-key": "arn:aws:kms:us-
west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
      }
    }
  ]
}
```

Als Nächstes folgt ein Beispiel für die Ablehnung von Versuchen, AWS KMS Schlüssel zu entfernen, die bereits einem Cluster zugeordnet sind.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
    ],
    "Resource": "*",
    "Condition": {
      "Null": {
        "ecs:fargate-ephemeral-storage-kms-key": "true"
```

```
      }
    }
  }
}
```

Kunden können mithilfe der Befehle, oder sehen, ob ihre nicht verwalteten Aufgaben oder Serviceaufgaben mithilfe des AWS CLI `describe-tasks` Schlüssels verschlüsselt sind. `describe-cluster` `describe-services`

Weitere Informationen finden Sie unter Bedingungsschlüssel für AWS KMS im AWS KMS Entwicklerhandbuch.

AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie in der linken Navigationsleiste Cluster und oben rechts entweder Cluster erstellen oder wählen Sie einen vorhandenen Cluster aus. Wählen Sie für einen vorhandenen Cluster oben rechts die Option Cluster aktualisieren aus.
3. Im Abschnitt Verschlüsselung des Workflows haben Sie die Möglichkeit, Ihren AWS KMS Schlüssel unter Managed Storage und Fargate Ephemeral Storage auszuwählen. Sie können von hier aus auch wählen, ob Sie einen AWS KMS Schlüssel erstellen möchten.
4. Wählen Sie Erstellen, wenn Sie mit der Erstellung Ihres neuen Clusters fertig sind, oder Aktualisieren, wenn Sie einen vorhandenen aktualisieren wollten.

AWS CLI

Im Folgenden finden Sie ein Beispiel für die Erstellung eines Clusters und die Konfiguration Ihres kurzlebigen Fargate-Speichers mithilfe von AWS CLI (ersetzen Sie die *roten* Werte durch Ihre eigenen):

```
aws ecs create-cluster --cluster clusterName \
--configuration '{"managedStorageConfiguration":
{"fargateEphemeralStorageKmsKeyId":"arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}}'
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:012345678901:cluster/clusterName",
        "clusterName": "clusterName",
```

```
        "configuration": {
            "managedStorageConfiguration": {
                "fargateEphemeralStorageKmsKeyId": "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": []
    },
    "clusterCount": 5
}
```

AWS CloudFormation

Im Folgenden finden Sie eine Beispielvorlage für die Erstellung eines Clusters und die Konfiguration Ihres kurzlebigen Fargate-Speichers mithilfe von AWS CloudFormation (ersetzen Sie die *roten* Werte durch Ihre eigenen):

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: "clusterName"
      Configuration:
        ManagedStorageConfiguration:
          FargateEphemeralStorageKmsKeyId: "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Verwaltung von AWS KMS Schlüsseln für den kurzlebigen Speicher von Fargate

Nachdem Sie Ihren AWS KMS Schlüssel zur Verschlüsselung Ihres kurzlebigen Fargate-Speichers erstellt oder importiert haben, verwalten Sie ihn genauso wie jeden anderen Schlüssel. AWS KMS

Automatische Rotation der Schlüssel AWS KMS

Sie können die automatische Schlüsselrotation aktivieren oder sie manuell drehen. Bei der automatischen Schlüsselrotation wird der Schlüssel jährlich für Sie rotiert, indem neues kryptografisches Material für den Schlüssel generiert wird. AWS KMS speichert auch alle früheren Versionen des kryptografischen Materials, sodass Sie alle Daten entschlüsseln können, für die die früheren Schlüsselversionen verwendet wurden. Jegliches rotiertes Material wird AWS KMS erst gelöscht, wenn Sie den Schlüssel löschen.

Die automatische Schlüsselrotation ist optional und kann jederzeit aktiviert oder deaktiviert werden.

Schlüssel deaktivieren oder widerrufen AWS KMS

Wenn Sie einen vom Kunden verwalteten Schlüssel deaktivieren AWS KMS, hat dies keine Auswirkungen auf die Ausführung von Aufgaben, und sie funktionieren während ihres gesamten Lebenszyklus weiter. Wenn eine neue Aufgabe den deaktivierten oder widerrufenen Schlüssel verwendet, schlägt die Aufgabe fehl, da sie nicht auf den Schlüssel zugreifen kann. Sie sollten einen CloudWatch Alarm oder ähnliches einrichten, um sicherzustellen, dass kein deaktivierter Schlüssel zum Entschlüsseln bereits verschlüsselter Daten benötigt wird.

Schlüssel löschen AWS KMS

Das Löschen von Schlüsseln sollte immer der letzte Ausweg sein und sollte nur durchgeführt werden, wenn Sie sicher sind, dass der gelöschte Schlüssel nie wieder benötigt wird. Neue Aufgaben, die versuchen, den gelöschten Schlüssel zu verwenden, schlagen fehl, weil sie nicht darauf zugreifen können. AWS KMS empfiehlt, einen Schlüssel zu deaktivieren, anstatt ihn zu löschen. Wenn Sie der Meinung sind, dass es notwendig ist, einen Schlüssel zu löschen, empfehlen wir, ihn zuerst zu deaktivieren und einen CloudWatch Alarm einzustellen, um sicherzustellen, dass er nicht benötigt wird. Wenn Sie einen Schlüssel löschen, haben Sie AWS KMS mindestens sieben Tage Zeit, Ihre Meinung zu ändern.

Überprüfung des AWS KMS Schlüsselzugriffs

Sie können CloudTrail Protokolle verwenden, um den Zugriff auf
Ihren AWS KMS Schlüssel zu überprüfen. Sie können die AWS KMS
Operationen`CreateGrant`,`GenerateDataKeyWithoutPlaintext`, und überprüfen`Decrypt`. Bei diesen Vorgängen wird auch das `aws:ecs:clusterAccount` und `aws:ecs:clusterName` als Teil der `EncryptionContext` angemeldeten Daten angezeigt CloudTrail.

Im Folgenden finden Sie CloudTrail Beispielereignisse für
`GenerateDataKeyWithoutPlaintextGenerateDataKeyWithoutPlaintext`

`(DryRun)`,`CreateGrant`,`CreateGrant (DryRun)`, und `RetireGrant` (ersetzen Sie die *roten* Werte durch Ihre eigenen).

GenerateDataKeyWithoutPlaintext


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 64,
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ebs:id": "vol-xxxxxxx",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
```

```
    "eventCategory": "Management"
}
```

GenerateDataKeyWithoutPlaintext (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "dryRun": true,
        "numberOfBytes": 64,
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
```

```
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ebs:id": "vol-xxxx",
                "aws:ecs:clusterName": "cluster-name"
            }
        },
        "retiringPrincipal": "ec2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
```

```
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```


CreateGrant (DryRun)


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "dryRun": true,
        "operations": [
            "Decrypt"
        ],
        "constraints": {
```

```
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ecs:clusterName": "cluster-name"
            }
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

RetireGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2024-04-20T18:37:38Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
```

```
    },
    "additionalEventData": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

AWS Häufig gestellte Fragen zur Fargate-Aufgabenwartung auf Amazon ECS

Was bedeutet Wartung und Stilllegung von Fargate-Aufgaben?

AWS ist für die Wartung der zugrunde liegenden Infrastruktur für AWS Fargate verantwortlich. AWS bestimmt, wann eine Revision einer Plattformversion durch eine neue Version für die Infrastruktur ersetzt werden muss. Dies wird als Ausmusterung von Aufgaben bezeichnet. AWS sendet eine Benachrichtigung über die Außerbetriebnahme einer Aufgabe, wenn die Revision einer Plattformversion zurückgezogen wird. Wir aktualisieren regelmäßig unsere unterstützten Plattformversionen, um eine neue Version einzuführen, die Updates für die Fargate-Runtime-Software und die zugrunde liegenden Abhängigkeiten wie das Betriebssystem und die Container-Laufzeit enthält. Sobald eine neuere Version verfügbar ist, ziehen wir die ältere Version zurück, um sicherzustellen, dass alle Kunden-Workloads auf der aktuellsten Version der Fargate-Plattformversion ausgeführt werden. Wenn eine Version außer Betrieb genommen wird, werden alle Aufgaben, die für diese Version ausgeführt werden, gestoppt.

Amazon-ECS-Aufgaben können entweder als Serviceaufgaben oder als eigenständige Aufgaben kategorisiert werden. Serviceaufgaben werden als Teil eines Service bereitgestellt und durch den Amazon ECS-Zeitplan gesteuert. Weitere Informationen finden Sie unter Amazon-ECS-Dienstleistungen. Eigenständige Aufgaben sind Aufgaben, die von der Amazon `RunTask` ECS-API entweder direkt oder von einem externen Scheduler gestartet werden, z. B. geplante Aufgaben (die von Amazon gestartet werden EventBridge) AWS Batch, oder AWS Step Functions.

Bei Serviceaufgaben müssen Sie keine Maßnahmen ergreifen, es sei denn, Sie möchten diese Aufgaben vorher AWS ersetzen. Wenn der Amazon ECS-Planer die Aufgaben beendet, verwendet er den minimalen fehlerfreien Prozentsatz und startet eine neue Aufgabe, um die gewünschte Anzahl für den Service aufrechtzuerhalten. Standardmäßig beträgt der minimale fehlerfreie Prozentsatz eines Services 100 Prozent, sodass eine neue Aufgabe zuerst gestartet wird, bevor eine Aufgabe beendet wird. Serviceaufgaben werden routinemäßig auf die gleiche Weise ersetzt, wenn Sie den Service skalieren, Konfigurationsänderungen vornehmen oder Überarbeitungen der Aufgabendefinitionen bereitstellen. Um sich auf den Prozess der Aufgaben-Außerbetriebnahme vorzubereiten, empfehlen wir, das Verhalten Ihrer Anwendung durch Simulieren dieses Szenarios zu testen. Halten Sie dazu eine Aufgabe in Ihrem Service an, um einen Test auf Ausfallsicherheit durchzuführen.

AWS Beendet bei der Außerbetriebnahme eigenständiger Aufgaben die Aufgabe am oder nach dem Außerbetriebnahmedatum der Aufgabe. Wir starten keine Ersatzaufgabe, wenn eine Aufgabe gestoppt wird. Wenn Sie möchten, dass diese Aufgaben weiterhin ausgeführt werden, müssen Sie die laufenden Aufgaben beenden und vor dem in der Benachrichtigung angegebenen Zeitpunkt eine Ersatzaufgabe starten. Daher empfehlen wir unseren Kunden, den Status der eigenständigen Aufgaben zu überwachen und bei Bedarf Logik zu implementieren, um die gestoppten Aufgaben zu ersetzen.

Wenn eine Aufgabe in einem der Szenarien gestoppt wird, können Sie `describe-tasks` ausführen. Der `stoppedReason` in der Antwort ist `ECS is performing maintenance on the underlying infrastructure hosting the task`.

Die Aufgabenwartung gilt, wenn eine neue Plattformversion durch eine neue Version ersetzt werden muss. Wenn es ein Problem mit einem zugrunde liegenden Fargate-Host gibt, ersetzt Amazon ECS den Host ohne Benachrichtigung über die Außerbetriebnahme der Aufgabe.

Was steht in der Mitteilung über die Außerbetriebnahme der Aufgabe?

Die Benachrichtigungen zur Einstellung der Aufgabe werden sowohl über das AWS Health Dashboard als auch per E-Mail an die registrierte E-Mail-Adresse gesendet und enthalten die folgenden Informationen:

- Das Datum der Außerbetriebnahme der Aufgabe – Die Aufgabe wird an oder nach diesem Datum beendet.
- Bei eigenständigen Aufgaben die IDs der Aufgaben.
- Bei Serviceaufgaben die ID des Clusters, auf dem der Service ausgeführt wird, und die IDs des Services.
- Die nächsten Schritte, die Sie unternehmen müssen.

In der Regel senden wir jeweils eine Benachrichtigung für Service- und eigenständige Aufgaben AWS-Region. In bestimmten Fällen erhalten Sie jedoch möglicherweise mehr als ein Ereignis für jeden Aufgabentyp, z. B. wenn zu viele Aufgaben nicht zurückgezogen werden müssen, wodurch die Grenzen unserer Benachrichtigungsmechanismen überschritten werden.

Es gibt folgende Möglichkeiten, Aufgaben zu identifizieren, die für die Außerbetriebnahme vorgesehen sind:

- Das AWS Health Dashboard

  AWS Health Benachrichtigungen können über Amazon EventBridge an Archivspeicher wie Amazon Simple Storage Service gesendet werden, automatisierte Aktionen wie das Ausführen einer AWS Lambda Funktion oder andere Benachrichtigungssysteme wie Amazon Simple Notification Service ausführen. Weitere Informationen finden Sie unter AWS Health Ereignisse mit Amazon überwachen EventBridge. Eine Beispielkonfiguration zum Senden von Benachrichtigungen an Amazon Chime, Slack oder Microsoft Teams finden Sie im AWS Health Aware-Repository unter. GitHub

  Das Folgende ist ein EventBridge Beispielereignis.

```
{
    "version": "0",
    "id": "3c268027-f43c-0171-7425-1d799EXAMPLE",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-08-16T23:18:51Z",
    "region": "us-east-1",
    "resources": [
        "cluster/service",
        "cluster/service"
    ],
```

```
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/ECS/
AWS_ECS_TASK_PATCHING_RETIREMENT/AWS_ECS_TASK_PATCHING_RETIREMENT_test1",
        "service": "ECS",
        "eventScopeCode": "ACCOUNT_SPECIFIC",
        "communicationId":
 "7988399e2e6fb0b905ddc88e0e2de1fd17e4c9fa60349577446d95a18EXAMPLE",
        "lastUpdatedTime": "Wed, 16 Aug 2023 23:18:52 GMT",
        "eventRegion": "us-east-1",
        "eventTypeCode": "AWS_ECS_TASK_PATCHING_RETIREMENT",
        "eventTypeCategory": "scheduledChange",
        "startTime": "Wed, 16 Aug 2023 23:18:51 GMT",
        "endTime": "Fri, 18 Aug 2023 23:18:51 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "\\nA software update has been deployed to
 Fargate which includes CVE patches or other critical patches. No action is required
 on your part. All new tasks launched automatically uses the latest software
 version. For existing tasks, your tasks need to be restarted in order for these
 updates to apply. Your tasks running as part of the following ECS Services will
 be automatically updated beginning Wed, 16 Aug 2023 23:18:51 GMT.\\n\\nAfter Wed,
 16 Aug 2023 23:18:51 GMT, the ECS scheduler will gradually replace these tasks,
 respecting the deployment settings for your service. Typically, services should
 see little to no interruption during the update and no action is required. When AWS
 stops tasks, AWS uses the minimum healthy percent (1) and launches a new task in
 an attempt to maintain the desired count for the service. By default, the minimum
 healthy percent of a service is 100 percent, so a new task is started first before
 a task is stopped. Service tasks are routinely replaced in the same way when
 you scale the service or deploy configuration changes or deploy task definition
 revisions. If you would like to control the timing of this restart you can update
 the service before Wed, 16 Aug 2023 23:18:51 GMT, by running the update-service
 command from the ECS command-line interface specifying force-new-deployment for
 services using Rolling update deployment type. For example:\\n\\n$ aws ecs update-
service -service service_name \\\n--cluster cluster_name -force-new-deployment\
\n\\nFor services using Blue/Green deployment type with AWS CodeDeploy:\\nPlease
 refer to create-deployment document (2) and create new deployment using same task
 definition revision.\\n\\nFor further details on ECS deployment types, please
 refer to ECS Deployment Developer Guide (1).\\nFor further details on Fargate's
 update process, please refer to the AWS Fargate User Guide (3).\\nIf you have
 any questions or concerns, please contact AWS Support (4).\\n\\n(1) https://
docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-types.html\\n(2)
 https://docs.aws.amazon.com/cli/latest/reference/deploy/create-deployment.html\\n(3)
 https://docs.aws.amazon.com/AmazonECS/latest/userguide/task-maintenance.html\\n(4)
```

```
 https://aws.amazon.com/support\\n\\nA list of your affected resources(s) can be
 found in the 'Affected resources' tab in the 'Cluster/ Service' format in the AWS
 Health Dashboard. \\n\\n"
            }
        ],
        "affectedEntities": [
            {
                "entityValue": "cluster/service"
            },
            {
                "entityValue": "cluster/service"
            }
        ]
    }
}
```

- Email

  Eine E-Mail wird an die registrierte E-Mail-Adresse für die AWS-Konto ID gesendet.

Kann ich die Wartezeit für die Außerbetriebnahme von Aufgaben ändern?

Sie können den Zeitpunkt konfigurieren, zu dem Fargate mit der Außerbetriebnahme der Aufgabe beginnt. Wählen Sie für Workloads, die eine sofortige Anwendung der Updates erfordern, die Einstellung „Sofort“ (`0`). Wenn Sie mehr Kontrolle benötigen, z. B. wenn eine Aufgabe nur während eines bestimmten Zeitfensters gestoppt werden kann, konfigurieren Sie die Optionen 7 Tage (7) oder 14 Tage (14).

Wir empfehlen Ihnen, eine kürzere Wartezeit zu wählen, damit Sie neuere Versionen der Plattformversionen früher erwerben können.

Konfigurieren Sie die Wartezeit, indem Sie das `put-account-setting-default` Programm ausführen oder `put-account-setting` als Root-Benutzer oder als Administratorbenutzer ausführen. Verwenden Sie die Option `fargateTaskRetirementWaitPeriod` für `name` und die Option `value`, die auf einen der folgenden Werte eingestellt ist:

- `0`- AWS sendet die Benachrichtigung und beginnt sofort, die betroffenen Aufgaben zurückzuziehen.
- 7- AWS sendet die Benachrichtigung und wartet 7 Kalendertage, bevor mit der Außerbetriebnahme der betroffenen Aufgaben begonnen wird.

- 14 – AWS sendet die Benachrichtigung und wartet 14 Kalendertage, bevor mit der Außerbetriebnahme der betroffenen Aufgaben begonnen wird.

Der Standardwert ist 7 Tage.

Weitere Informationen finden Sie unter put-account-setting-default und put-account-setting in der Referenz zu Amazon Elastic Container Service API.

Weitere Informationen finden Sie unter AWS Fargate Aufgabe, Stilllegung, Wartezeit.

Kann ich über andere AWS Dienste Benachrichtigungen über die Einstellung von Aufgaben erhalten?

AWS sendet eine Benachrichtigung über die Einstellung einer Aufgabe an den AWS Health Dashboard und an den primären E-Mail-Kontakt auf der AWS-Konto. Das AWS Health Dashboard bietet eine Reihe von Integrationen in andere AWS Dienste, darunter EventBridge. Sie können EventBridge damit die Sichtbarkeit der Hinweise automatisieren (z. B. die Nachricht an ein ChatOps Tool weiterleiten). Weitere Informationen finden Sie unter Lösungsübersicht: Benachrichtigungen über die Einstellung von Aufgaben erfassen.

Kann ich die Außerbetriebnahme einer Aufgabe ändern, nachdem sie geplant wurde?

Nein. Der Zeitplan basiert auf der Wartezeit für die Außerbetriebnahme von Aufgaben, die standardmäßig auf 7 Tage eingestellt ist. Wenn Sie mehr Zeit benötigen, können Sie die Wartezeit auf 14 Tage konfigurieren. Weitere Informationen finden Sie unter Kann ich die Wartezeit für die Außerbetriebnahme von Aufgaben ändern?. Die Änderung dieser Konfiguration gilt für Pensionierungen, die in der future geplant sind. Derzeit geplante Stilllegungen sind davon nicht betroffen. Wenn Sie weitere Bedenken haben, wenden Sie sich an. AWS Support

Kann ich den Zeitpunkt der Ersetzung einer Aufgabe kontrollieren?

Bei Diensten, die die fortlaufende Bereitstellung verwenden, aktualisieren Sie den Dienst `update-service` mithilfe der `force-deployment` Option vor dem Start der Außerbetriebnahme.

Im folgenden `update-service` Beispiel wird die `force-deployment` Option verwendet.

```
aws ecs update-service –-service service_name \
    --cluster cluster_name \
```

```
    --force-new-deployment
```

Für Dienste, die die blaue/grüne Bereitstellung verwenden, müssen Sie eine neue Bereitstellung in AWS CodeDeploy erstellen. Informationen zum Erstellen der Bereitstellung finden Sie unter Create-Deployment in der Referenz.AWS Command Line Interface

Wie geht Amazon ECS mit Aufgaben um, die Teil eines Service sind?

Amazon ECS ersetzt nach und nach die betroffenen Aufgaben in Ihrem Service, wenn die Pensionierung von Fargate beginnt. Wenn Amazon ECS eine Aufgabe beendet, verwendet es den minimalen fehlerfreien Prozentsatz des Service und startet eine neue Aufgabe, um die gewünschte Anzahl an Aufgaben für den Service beizubehalten. Eine neue Aufgabe wird gestartet, bevor eine Aufgabe gestoppt wird, da der standardmäßige Mindeststatus bei 100 liegt. Dienstaufgaben werden routinemäßig auf die gleiche Weise ersetzt, wenn Sie den Dienst skalieren, Konfigurationsänderungen vornehmen oder Überarbeitungen der Aufgabendefinitionen bereitstellen. Weitere Informationen zum fehlerfreien Mindestprozentsatz finden Sie unter. Bereitstellungskonfiguration

Kann Amazon ECS eigenständige Aufgaben automatisch bearbeiten?

Nein. AWS kann keine Ersatzaufgabe für eigenständige Aufgaben erstellen`RunTask`, die von geplanten Aufgaben (z. B. über EventBridge Scheduler) AWS Batch, oder AWS Step Functions gestartet werden. Amazon ECS verwaltet nur Aufgaben, die Teil eines Service sind.

Unterstützte Regionen für Amazon ECS auf AWS Fargate

Sie können die folgenden Tabellen verwenden, um die Regionsunterstützung für Linux-Container auf AWS Fargate und Windows-Container auf AWS Fargate zu überprüfen.

Linux-Container auf AWS Fargate

Amazon ECS Linux-Container auf AWS Fargate werden im Folgenden unterstützt AWS-Regionen. Die unterstützten Availability Zone-IDs werden gegebenenfalls notiert.

Name der Region	Region
USA Ost (Ohio)	us-east-2
USA Ost (Nord-Virginia)	us-east-1

Name der Region	Region
USA West (Nordkalifornien)	us-west-1 (nur `usw1-az1` und `usw1-az3`)
USA West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Tokio)	ap-northeast-1 (nur `apne1-az1` , `apne1-az2` und `apne1-az4`)
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Melbourne)	ap-southeast-4
Kanada (Zentral)	ca-central-1
Kanada West (Calgary)	ca-west-1
China (Peking)	cn-north-1 (nur `cnn1-az1` und `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zürich)	eu-central-2
Europa (Irland)	eu-west-1

Name der Region	Region
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Mailand)	eu-south-1
Europa (Spanien)	eu-south-2
Europa (Stockholm)	eu-north-1
Südamerika (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Naher Osten (Bahrain)	me-south-1
Naher Osten (VAE)	me-central-1
AWS GovCloud (USA-Ost)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

Windows-Container auf AWS Fargate

Amazon ECS Windows-Container auf AWS Fargate werden im Folgenden unterstützt AWS-Regionen. Die unterstützten Availability Zone-IDs werden gegebenenfalls notiert.

Name der Region	Region
USA Ost (Ohio)	us-east-2
USA Ost (Nord-Virginia)	us-east-1 (nur `use1-az1`, `use1-az2`, `use1-az4`, `use1-az5` und `use1-az6`)
USA West (Nordkalifornien)	us-west-1 (nur `usw1-az1` und `usw1-az3`)
USA West (Oregon)	us-west-2

Name der Region	Region
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Melbourne)	ap-southeast-4
Asien-Pazifik (Tokio)	ap-northeast-1 (nur `apne1-az1` , `apne1-az2` und `apne1-az4`)
Kanada (Zentral)	ca-central-1 (nur `cac1-az1` und `cac1-az2`)
Kanada West (Calgary)	ca-west-1
China (Peking)	cn-north-1 (nur `cnn1-az1` und `cnn1-az2`)
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zürich)	eu-central-2
Europa (Irland)	eu-west-1
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Mailand)	eu-south-1

Name der Region	Region
Europa (Spanien)	eu-south-2
Europa (Stockholm)	eu-north-1
Südamerika (São Paulo)	sa-east-1
Israel (Tel Aviv)	il-central-1
Naher Osten (VAE)	me-central-1
Naher Osten (Bahrain)	me-south-1

Entwickeln Sie Ihre Lösung für Amazon ECS

Bevor Sie Amazon ECS verwenden, müssen Sie Entscheidungen über Kapazität, Netzwerk, Kontoeinstellungen und Protokollierung treffen, damit Sie Ihre Amazon ECS-Ressourcen korrekt konfigurieren können.

Capacity (Kapazität)

Die Kapazität ist die Infrastruktur, in der Ihre Container laufen. Im Folgenden sind die Optionen aufgeführt:

- Amazon EC2-Instances
- Serverlos ()AWS Fargate (Fargate)
- On-Premises-virtuelle-Maschinen (VM) oder -Server

Sie geben die Infrastruktur an, wenn Sie einen Cluster erstellen. Sie geben auch den Infrastrukturtyp an, wenn Sie eine Aufgabendefinition registrieren. In der Aufgabendefinition wird die Infrastruktur als „Starttyp“ bezeichnet. Sie verwenden den Starttyp auch, wenn Sie eine eigenständige Aufgabe ausführen oder einen Dienst bereitstellen. Informationen zu den Starttyp-Optionen finden Sie unterAmazon-ECS-Starttypen.

Netzwerk

AWS Ressourcen werden in Subnetzen erstellt. Wenn Sie EC2-Instances verwenden, startet Amazon ECS die Instances in dem Subnetz, das Sie bei der Erstellung eines Clusters angeben. Ihre Aufgaben werden im Instance-Subnetz ausgeführt. Für Fargate oder lokale virtuelle Maschinen geben Sie das Subnetz an, wenn Sie eine Aufgabe ausführen oder einen Dienst erstellen.

Je nach Ihrer Anwendung kann es sich bei dem Subnetz um ein privates oder öffentliches Subnetz handeln, und das Subnetz kann sich in einer der folgenden Ressourcen befinden: AWS

- Availability Zones
- Local Zones
- Wavelength-Zonen
- AWS-Regionen
- AWS Outposts

Weitere Informationen finden Sie unter Amazon ECS-Anwendungen in gemeinsam genutzten Subnetzen, Local Zones und Wellenlängenzonen oder Amazon Elastic Container Service auf AWS Outposts.

Sie können Ihre Anwendung mithilfe einer der folgenden Methoden mit dem Internet verbinden:

- Ein öffentliches Subnetz mit einem Internet-Gateway

  Verwenden Sie öffentliche Subnetze, wenn Sie öffentliche Anwendungen haben, die viel Bandbreite oder minimale Latenz benötigen. Zu den anwendbaren Szenarien gehören Videostreaming- und Spieledienste.
- Ein privates Subnetz mit einem NAT-Gateway

  Verwenden Sie private Subnetze, wenn Sie Ihre Container vor direktem externen Zugriff schützen möchten. Zu den anwendbaren Szenarien gehören Zahlungsabwicklungssysteme oder Container, in denen Benutzerdaten und Passwörter gespeichert werden.

Zugriff auf Funktionen

Sie können Ihre Amazon ECS-Kontoeinstellungen verwenden, um auf die folgenden Funktionen zuzugreifen:

- Container Insights

  CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Die Metriken umfassen die Auslastung für Ressourcen wie z B. CPU, Arbeitsspeicher, Datenträger und Netzwerk.
- `awsvpc`Bündelung

  Für bestimmte EC2-Instance-Typen können zusätzliche Netzwerkschnittstellen (ENIs) auf neu gestarteten Container-Instances verfügbar sein.
- Tagging-Autorisierung

  Benutzer müssen über Berechtigungen für Aktionen verfügen, mit denen eine Ressource erstellt wird, z. B. `ecsCreateCluster` Wenn bei der Aktion zur Erstellung einer Ressource Tags angegeben wurden, AWS führt dieser Vorgang eine zusätzliche Autorisierung durch, um zu überprüfen, ob Benutzer oder Rollen berechtigt sind, Tags zu erstellen. `ecs:TagResource`
- FIPS-140-Compliance von Fargate

Fargate unterstützt den Bundesstandard für Informationsprozesse (FIPS-140), der die Sicherheitsanforderungen für Verschlüsselungsmodule zum Schutz sensibler Daten festlegt. Es ist der aktuelle Standard der amerikanischen und kanadischen Regierung und gilt für Systeme, die mit dem Federal Information Security Management Act (FISMA) oder dem Federal Risk and Authorization Management Program (FedRAMP) konform sein müssen.

- Änderungen der Ruhestandszeit für Fargate-Aufgaben

  Sie können die Wartezeit konfigurieren, bis Fargate-Aufgaben für das Patchen zurückgezogen werden.

- Dual-Stack-VPC

  Erlauben Sie Aufgaben, über IPv4, IPv6 oder beides zu kommunizieren.

- Format des Amazon-Ressourcennamens (ARN)

  Bestimmte Funktionen, wie z. B. die Tagging-Autorisierung, erfordern ein neues Amazon Resource Name (ARN) -Format.

Weitere Informationen finden Sie unter Greifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu.

IAM-Rollen

Eine IAM-Rolle ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. In Amazon ECS können Sie Rollen erstellen, um Berechtigungen für Amazon ECS-Ressourcen wie Container oder Services zu erteilen.

Für einige Amazon ECS-Funktionen sind Rollen erforderlich. Weitere Informationen finden Sie unter IAM-Rollen für Amazon ECS.

Protokollierung

Protokollierung und Überwachung sind wichtige Aspekte zur Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon ECS-Workloads. Verfügbar sind die nachfolgend aufgeführten Optionen:

- Amazon CloudWatch Logs — Logs weiterleiten an Amazon CloudWatch

- FireLensfür Amazon ECS — leitet Protokolle zur Speicherung und Analyse von Protokollen an einen AWS Service oder ein AWS Partner Network Ziel weiter. The AWS Partner Network ist eine globale Partnergemeinschaft, die Programme, Fachwissen und Ressourcen nutzt, um Kundenangebote aufzubauen, zu vermarkten und zu verkaufen.

Amazon-ECS-Starttypen

Der Starttyp der Aufgabendefinition definiert beispielsweise, mit welcher Kapazität die Aufgabe ausgeführt werden kann AWS Fargate.

Nachdem Sie den Starttyp ausgewählt haben, überprüft Amazon ECS, ob die von Ihnen konfigurierten Aufgabendefinitionsparameter mit dem Starttyp funktionieren.

Fargate

Fargate ist eine serverlose pay-as-you-go Compute-Engine, mit der Sie sich auf die Erstellung von Anwendungen konzentrieren können, ohne Server verwalten zu müssen. Wenn Sie sich für Fargate entscheiden, müssen Sie keine EC2-Infrastruktur verwalten. Sie müssen lediglich Ihr Container-Image erstellen und definieren, auf welchem Cluster Sie Ihre Anwendungen ausführen möchten. Fargate verfügt über eine native Integration mit AWS Diensten wie:

- Amazon VPC
- Auto Scaling
- Elastic Load Balancing
- IAM
- Secrets Manager

Mit Fargate haben Sie mehr Kontrolle als mit EC2, da Sie genau die CPU und den Arbeitsspeicher auswählen, die Ihre Anwendung benötigt. Fargate kümmert sich um die Skalierung Ihrer Kapazität, sodass Sie sich keine Gedanken über Verkehrsspitzen machen müssen. Das bedeutet, dass der operative Aufwand bei Fargate geringer ist.

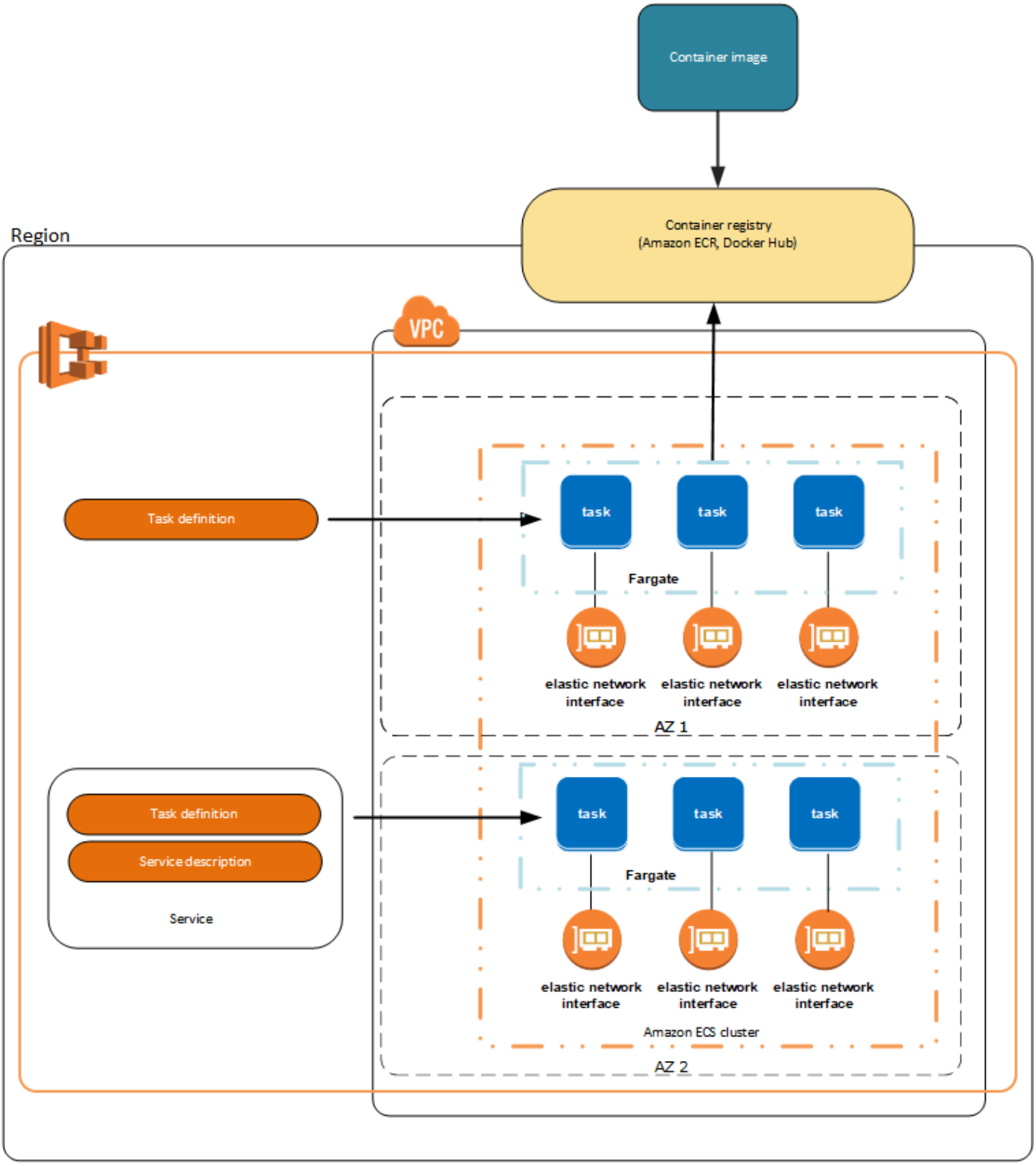
Fargate erfüllt die Standards für Compliance-Programme wie PCI, FIPS 140-2, FedRAMP und HIPAA. Weitere Informationen finden Sie unter Services in Scope by Compliance Program.AWS

Fargate eignet sich für die folgenden Workloads:

- Große Workloads, die einen geringen Betriebsaufwand erfordern
- Kleine Workloads mit gelegentlichen Spitzenleistungen
- Winzige Workloads
- Batch-Workloads

Weitere Informationen über Regionen, die Fargate unterstützen, finden Sie unter the section called “AWS Fargate-Regionen”.

Das folgende Diagramm zeigt die allgemeine Architektur.



Weitere Informationen zu Amazon ECS auf Fargate finden Sie unter [AWS Fargatefür Amazon ECS](#).

EC2

Der EC2-Starttyp eignet sich für große Workloads, die preisoptimiert werden müssen.

Wenn Sie darüber nachdenken, wie Sie Aufgabendefinitionen und Services unter Verwendung des Starttyps EC2 abbilden sollen, empfehlen wir Ihnen, zu überlegen, welche Prozesse zusammen auf der gleichen Instance ausgeführt werden müssen und wie jede Komponente skaliert werden soll.

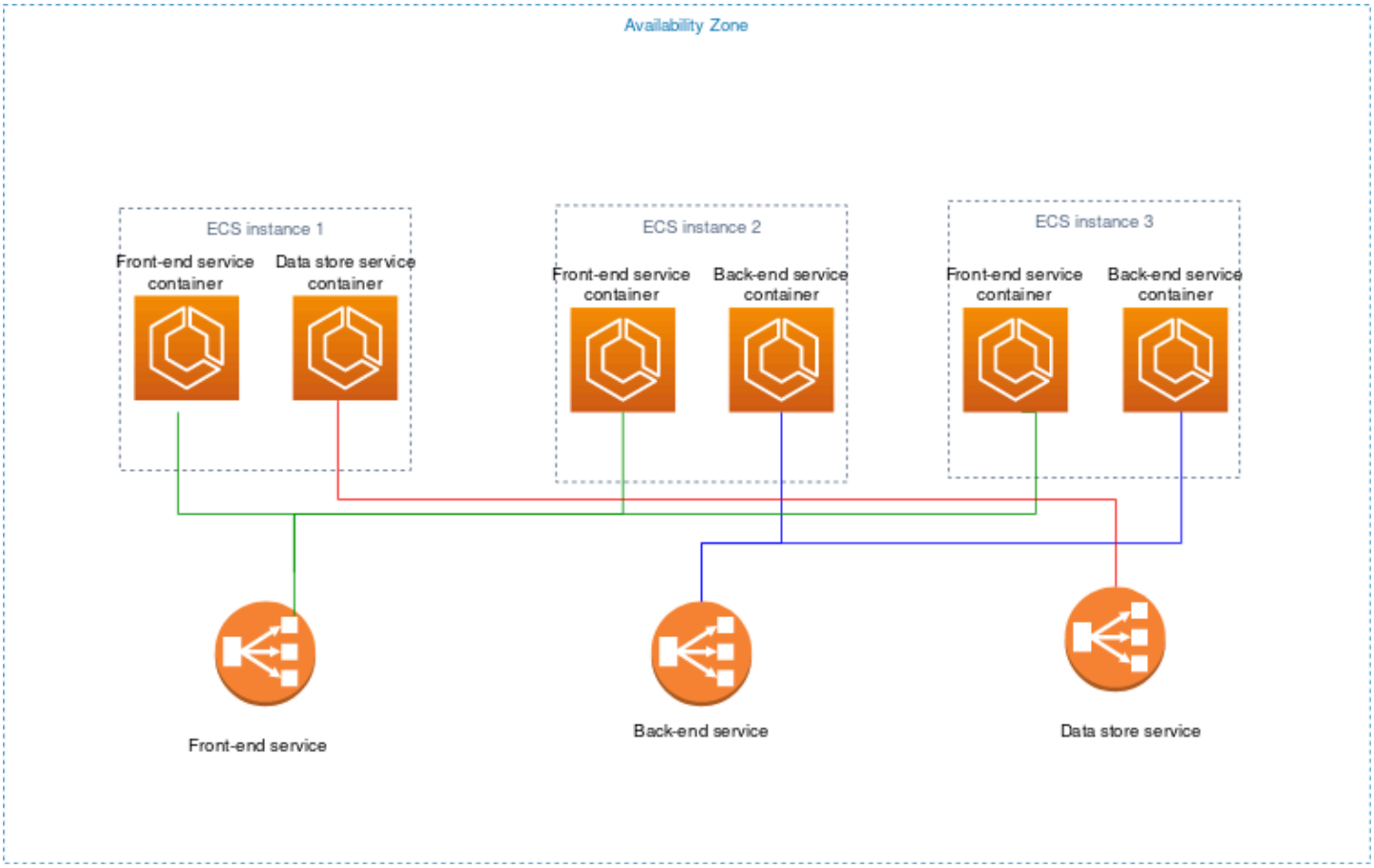
Stellen Sie sich z. B. eine Anwendung vor, die aus folgenden Komponenten besteht:

- Ein Front-End-Service, der Informationen auf einer Webseite anzeigt
- Ein Backend-Service, der APIs für den Front-End-Service bereitstellt
- Ein Datenspeicher

Erstellen Sie für dieses Beispiel Aufgabendefinitionen, die die Container gruppieren, die für einen gemeinsamen Zweck verwendet werden. Teilen Sie die verschiedenen Komponenten in mehrere und separate Aufgabendefinitionen auf. Der folgende Beispielcluster verfügt über drei Container-Instances, in denen drei Front-End-Service-Container, zwei Backend-Service-Container und ein Datenspeicher-Service-Container ausgeführt werden.

Sie können verwandte Container in einer Aufgabendefinition gruppieren, z. B. verknüpfte Container, die gemeinsam ausgeführt werden müssen. Sie könnten beispielsweise einen Protokoll-Streaming-Container zu Ihrem Front-End-Service hinzufügen und dies in die gleiche Aufgabendefinition aufnehmen.

Nachdem Sie die Aufgabendefinitionen fertiggestellt haben, können Sie auf dieser Grundlage Services erstellen, um die Verfügbarkeit Ihrer gewünschten Aufgaben sicherzustellen. Weitere Informationen finden Sie unter Einen Amazon ECS-Service mithilfe der Konsole erstellen. In Ihren Services können Sie Container mit Elastic Load Balancing-Load Balancern verknüpfen. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen. Wenn sich Ihre Anwendungsanforderungen ändern, können Sie Ihre Service aktualisieren, um die Anzahl der gewünschten Aufgaben nach oben oder unten zu skalieren. Oder Sie können Ihre Services aktualisieren, um neuere Versionen der Container in Ihren Aufgaben bereitzustellen. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.



Extern

Der externe Starttyp wird verwendet, um Ihre containerisierten Anwendungen auf Ihrem On-Premises-Server oder virtuellen Computer (VM) auszuführen, den Sie bei Ihrem Amazon-ECS-Cluster registrieren und remote verwalten. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den externen Starttyp.

Amazon ECS-Anwendungen in gemeinsam genutzten Subnetzen, Local Zones und Wellenlängenzonen

Amazon ECS unterstützt Workloads, die Local Zones und Wavelength Zones verwenden, und AWS Outposts für Anwendungen, bei denen geringe Latenz oder lokale Datenverarbeitung erforderlich sind.

- Sie können Local Zones als Erweiterung verwenden AWS-Region , um Ressourcen an mehreren Standorten näher an Ihren Endbenutzern zu platzieren.

- Sie können Wavelength Zones verwenden, um Anwendungen zu erstellen, die extrem niedrige Latenzen für 5G-Geräte und Endbenutzer bereitstellen. Wavelength stellt AWS Standard-Rechen- und Speicherdienste am Rand der 5G-Netzwerke von Telekommunikationsanbietern bereit.
- AWS Outposts bietet native Infrastruktur AWS-Services- und Betriebsmodelle für praktisch jedes Rechenzentrum, jeden Kollokationsraum oder jede Einrichtung vor Ort.

> **Important**
>
> Amazon AWS Fargate ECS-On-Workloads werden derzeit in Local Zones, Wavelength Zones oder on AWS Outposts nicht unterstützt.

Informationen zu den Unterschieden zwischen Local Zones, Wavelength Zones und AWS Outposts finden Sie in den AWS Wavelength FAQs unter Wie sollte ich darüber nachdenken, wann ich AWS Wavelength, AWS Local Zones oder AWS Outposts für Anwendungen verwenden sollte, die eine geringe Latenz oder lokale Datenverarbeitung erfordern.

Gemeinsam genutzte Subnetze

Sie können VPC-Freigabe nutzen, um Subnetze mit anderen AWS -Konten innerhalb derselben AWS Organizations zu teilen.

Sie können gemeinsam genutzte VPCs für den EC2-Starttyp verwenden, wobei die folgenden Überlegungen zu beachten sind:

- Der Besitzer des VPC-Subnetzes muss ein Subnetz mit einem Teilnehmerkonto teilen, bevor dieses Konto es für Amazon ECS-Ressourcen verwenden kann.
- Sie können die VPC-Standardsicherheitsgruppe nicht für Ihre Container-Instances verwenden, da sie dem Besitzer gehört. Darüber hinaus können Teilnehmer keine Instances mithilfe von Sicherheitsgruppen starten, die anderen Teilnehmern oder dem Eigentümer gehören.
- In einem gemeinsam genutzten Subnetz kontrollieren der Teilnehmer und der Eigentümer die Sicherheitsgruppen innerhalb des jeweiligen Kontos separat. Der Subnetzbesitzer kann von den Teilnehmern erstellte Sicherheitsgruppen zwar anzeigen, jedoch keine Aktionen bei diesen durchführen. Wenn der Subnetzbesitzer diese Sicherheitsgruppen entfernen oder ändern möchte, muss der Teilnehmer, der die Sicherheitsgruppe erstellt hat, die Aktion durchführen.
- Der Eigentümer einer gemeinsam genutzten VPC kann einen Cluster, den ein Teilnehmer im gemeinsam genutzten Subnetz erstellt, nicht anzeigen, aktualisieren oder löschen. Dies

gilt zusätzlich zu den VPC-Ressourcen, auf die jedes Konto unterschiedlich zugreifen kann. Weitere Informationen finden Sie unter Verantwortlichkeiten und Berechtigungen für Besitzer und Teilnehmer im Amazon-VPC-Benutzerhandbuch.

Sie können gemeinsam genutzte VPCs für den Fargate-Starttyp verwenden, wobei die folgenden Überlegungen zu beachten sind:

- Der Besitzer des VPC-Subnetzes muss ein Subnetz mit einem Teilnehmerkonto teilen, bevor dieses Konto es für Amazon ECS-Ressourcen verwenden kann.
- Sie können mit der Standardsicherheitsgruppe für die VPC keinen Dienst erstellen oder eine Aufgabe ausführen, da sie dem Besitzer gehört. Darüber hinaus können Teilnehmer mithilfe von Sicherheitsgruppen, die anderen Teilnehmern oder dem Besitzer gehören, keinen Dienst erstellen oder eine Aufgabe ausführen.
- In einem gemeinsam genutzten Subnetz kontrollieren der Teilnehmer und der Eigentümer die Sicherheitsgruppen innerhalb des jeweiligen Kontos separat. Der Subnetzbesitzer kann von den Teilnehmern erstellte Sicherheitsgruppen zwar anzeigen, jedoch keine Aktionen bei diesen durchführen. Wenn der Subnetzbesitzer diese Sicherheitsgruppen entfernen oder ändern möchte, muss der Teilnehmer, der die Sicherheitsgruppe erstellt hat, die Aktion durchführen.
- Der Eigentümer einer gemeinsam genutzten VPC kann einen Cluster, den ein Teilnehmer im gemeinsam genutzten Subnetz erstellt, nicht anzeigen, aktualisieren oder löschen. Dies gilt zusätzlich zu den VPC-Ressourcen, auf die jedes Konto unterschiedlich zugreifen kann. Weitere Informationen finden Sie unter Verantwortlichkeiten und Berechtigungen für Besitzer und Teilnehmer im Amazon-VPC-Benutzerhandbuch.

Weitere Informationen zur Freigabe von VPC-Subnetzen finden Sie unter Freigeben Ihrer VPC für andere Konten im Amazon-VPC-Benutzerhandbuch.

Local Zones

Eine lokale Zone ist eine Erweiterung einer Zone, die sich AWS-Region in unmittelbarer geografischer Nähe zu Ihren Benutzern befindet. Local Zones haben ihre eigenen Verbindungen mit dem Internet und unterstützen AWS Direct Connect. Ressourcen, die in einer Local Zone erstellt werden, können von lokalen Benutzern mit geringer Latenz genutzt werden. Weitere Informationen finden Sie unter AWS Local Zones.

Eine Local Zone wird durch einen Regionscode dargestellt, gefolgt von einer Kennung, die den Standort angibt, (z. B. `us-west-2-lax-1a`).

Um eine Local Zone verwenden zu können, müssen Sie sich für die Zone anmelden. Nach der Anmeldung, müssen Sie eine Amazon VPC und ein Subnetz in der Local Zone erstellen.

Sie können Amazon-EC2-Instances, Amazon-FSx-Dateiserver und Application Load Balancer für Ihre Amazon-ECS-Cluster und -Aufgaben starten.

Weitere Informationen finden Sie unter Was sind AWS Local Zones? im AWS Local Zones User Guide.

Wavelength Zones

Mit AWS Wavelength können Sie Anwendungen mit extrem niedriger Latenz für Mobilgeräte und Endbenutzer erstellen. Wavelength stellt AWS Standard-Rechen- und Speicherdienste am Rand der 5G-Netzwerke von Telekommunikationsanbietern bereit. Sie können eine Amazon Virtual Private Cloud auf eine oder mehrere Wavelength Zones erweitern. Anschließend können Sie AWS Ressourcen wie Amazon EC2 EC2-Instances verwenden, um Anwendungen auszuführen, die eine extrem niedrige Latenz und eine Verbindung zu der AWS-Services Region erfordern.

Eine Wavelength Zone ist eine isolierte Zone am Standort des Anbieters, in der die Wavelength-Infrastruktur bereitgestellt wird. Wellenlängenzonen sind an eine gebunden AWS-Region. Eine Wavelength-Zone ist eine logische Erweiterung einer Region und wird von der Steuerungsebene in der Region verwaltet.

Eine Wavelength Zone wird durch einen Regionscode gefolgt von einer Kennung dargestellt, die die Wavelength Zone angibt, (z. B. `us-east-1-wl1-bos-wlz-1`).

Um eine Wavelength Zone verwenden zu können, müssen Sie sich für die Zone anmelden. Nach der Anmeldung müssen Sie eine Amazon VPC und ein Subnetz in der Wavelength Zone erstellen. Anschließend können Sie Ihre Amazon EC2-Instances in der Zone starten, um sie für Ihre Amazon ECS-Cluster und -Aufgaben zu verwenden.

Weitere Informationen finden Sie unter Erste Schritte mit AWS Wavelength im AWS Wavelength - Entwicklerhandbuch.

Wellenlängenzonen sind nicht in allen verfügbar AWS-Regionen. Weitere Informationen zu den Regionen, die Wavelength Zones unterstützen, finden Sie unter Verfügbare Wavelength Zones im AWS Wavelength Developerhandbuch.

Amazon Elastic Container Service auf AWS Outposts

AWS Outposts ermöglicht native AWS Dienste, Infrastrukturen und Betriebsmodelle in lokalen Einrichtungen. In AWS Outposts Umgebungen können Sie dieselben AWS APIs, Tools und Infrastrukturen verwenden wie in der AWS Cloud.

Amazon ECS on AWS Outposts ist ideal für Workloads mit niedriger Latenz, die in unmittelbarer Nähe zu lokalen Daten und Anwendungen ausgeführt werden müssen.

Weitere Informationen zu AWS Outposts finden Sie im AWS Outposts Benutzerhandbuch.

Überlegungen

Im Folgenden finden Sie Überlegungen zur Verwendung von Amazon ECS auf AWS Outposts:

- Amazon Elastic Container Registry und Network Load Balancer werden in der AWS Region ausgeführt, nicht auf AWS Outposts. AWS Identity and Access Management Dies erhöht die Latenzen zwischen diesen Diensten und den Containern.
- AWS Fargate ist nicht verfügbar am. AWS Outposts

Im Folgenden finden Sie Überlegungen zur Netzwerkkonnektivität für AWS Outposts:

- Wenn die Netzwerkkonnektivität zwischen Ihrer AWS Region AWS Outposts und der Region unterbrochen wird, laufen Ihre Cluster weiter. Sie können jedoch keine neuen Cluster erstellen oder neue Aktionen für vorhandene Cluster ausführen, bis die Verbindung wiederhergestellt wurde. Bei Instance-Fehlern wird die Instance nicht automatisch ersetzt. Der CloudWatch Logs-Agent kann keine Protokolle und Ereignisdaten aktualisieren.
- Wir empfehlen Ihnen, eine zuverlässige, hochverfügbare Konnektivität mit geringer Latenz zwischen Ihrer Region AWS Outposts und AWS der Region bereitzustellen.

Voraussetzungen

Die folgenden Voraussetzungen gelten für die Verwendung von Amazon ECS auf AWS Outposts:

- Sie müssen ein Outpost in Ihrem On-Premises-Rechenzentrum installiert und konfiguriert haben.
- Sie müssen über eine zuverlässige Netzwerkverbindung zwischen Ihrem Outpost und der entsprechenden AWS -Region verfügen.

Erstellen Sie einen Cluster auf AWS Outposts

Um einen Amazon ECS-Cluster auf einem AWS Outposts mit dem zu erstellen AWS CLI, geben Sie eine Sicherheitsgruppe und ein Subnetz an, das mit Ihrem AWS Outposts verknüpft werden soll.

Um ein Subnetz zu erstellen, das Ihrem zugeordnet ist. AWS Outposts

```
aws ec2 create-subnet \
    --cidr-block 10.0.3.0/24 \
    --vpc-id vpc-xxxxxxxx \
    --outpost-arn arn:aws:outposts:us-west-2:123456789012:outpost/op-xxxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

Im folgenden Beispiel wird ein Amazon ECS-Cluster in einem AWS Outposts erstellt.

1. Erstellen Sie eine Rolle und eine Richtlinie mit aktivierten Rechten AWS Outposts.

   Die `role-policy.json`-Datei ist das Richtliniendokument, das die Auswirkungen und Aktionen für Ressourcen enthält. Informationen zum Dateiformat finden Sie unter PutRoleRichtlinie in der IAM-API-Referenz

   ```
   aws iam create-role --role-name ecsRole \
       --assume-role-policy-document file://ecs-policy.json
   aws iam put-role-policy --role-name ecsRole --policy-name ecsRolePolicy \
       --policy-document file://role-policy.json
   ```

2. Erstellen Sie ein IAM-Instance-Profil mit Rechten im AWS Outposts.

   ```
   aws iam create-instance-profile --instance-profile-name outpost
   aws iam add-role-to-instance-profile --instance-profile-name outpost \
       --role-name ecsRole
   ```

3. Erstellen Sie eine VPC.

   ```
   aws ec2 create-vpc --cidr-block 10.0.0.0/16
   ```

4. Erstellen Sie eine Sicherheitsgruppe für die Container-Instances und geben Sie den richtigen CIDR-Bereich für den AWS Outposts an. (Dieser Schritt ist anders für AWS Outposts.)

   ```
   aws ec2 create-security-group --group-name MyOutpostSG
   aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
   ```

```
    --port 22 --cidr 10.0.3.0/24
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 80 --cidr 10.0.3.0/24
```

5. Erstellen Sie den Cluster.
6. Definieren Sie die Umgebungsvariablen des Amazon-ECS-Container-Agenten, um die Instance in dem im vorherigen Schritt erstellten Cluster zu launchen und definieren Sie alle Tags, die Sie hinzufügen möchten, um den Cluster zu identifizieren (z. B. `Outpost`, um anzugeben, dass der Cluster für einen Outpost bestimmt ist).

   ```
   #! /bin/bash
   cat << 'EOF' >> /etc/ecs/ecs.config
   ECS_CLUSTER=MyCluster
   ECS_IMAGE_PULL_BEHAVIOR=prefer-cached
   ECS_CONTAINER_INSTANCE_TAGS={"environment": "Outpost"}
   EOF
   ```

   > **Note**
   >
   > Um Verzögerungen zu vermeiden, die durch das Abrufen von Container-Images aus Amazon ECR in die Region verursacht werden, verwenden Sie Image-Caches. Konfigurieren Sie dazu bei jeder Ausführung einer Aufgabe den Amazon ECS-Agenten so, dass er standardmäßig das zwischengespeicherte Image für die Instance selbst verwendet, indem Sie `ECS_IMAGE_PULL_BEHAVIOR` auf `prefer-cached` einstellen.

7. Erstellen Sie die Container-Instance, wobei Sie die VPC und das Subnetz für den AWS Outposts angeben, auf dem diese Instance ausgeführt werden soll, sowie einen Instance-Typ, der im AWS Outposts verfügbar ist. (Dieser Schritt ist anders für AWS Outposts.)

   Die `userdata.txt`-Datei enthält die Benutzerdaten, die die Instance für allgemeine automatisierte Konfigurationsaufgaben und sogar zur Ausführung von Skripts nach dem Start der Instance verwenden können. Informationen zur Datei für API-Aufrufe finden Sie unter Befehle auf Ihrer Linux-Instance beim Start ausführen im Amazon EC2 EC2-Benutzerhandbuch.

   ```
   aws ec2 run-instances --count 1 --image-id ami-xxxxxxxx --instance-type c5.large \
       --key-name aws-outpost-key --subnet-id subnet-xxxxxxxxxxxxxxxxx \
       --iam-instance-profile Name outpost --security-group-id sg-xxxxxx \
       --associate-public-ip-address --user-data file://userdata.txt
   ```

> **Note**
>
> Dieser Befehl wird auch verwendet, wenn dem Cluster zusätzliche Instances hinzugefügt werden. Alle Container, die im Cluster bereitgestellt werden, werden in diesem bestimmten AWS Outposts platziert.

8. Anmelden Ihrer Aufgabendefinition. Verwenden Sie den folgenden Befehl und ersetzen Sie `ecs-task.json` mit dem Namen Ihrer Aufgabendefinition.

```
aws ecs register-task-definition --cli-input-json file://ecs-task.json
```

9. Führen Sie die Aufgabe aus, oder erstellen Sie den Service.

   Run the task

```
aws ecs run-task --cluster mycluster --count 1 --task-definition outpost-app:1
```

   Create the service

```
aws ecs create-service --cluster mycluster --service-name outpost-service \
    --task-definition outpost-app:1 --desired-count 1
```

Optimieren Sie die Kapazität und Verfügbarkeit von Amazon ECS

Die Verfügbarkeit von Anwendungen ist entscheidend für ein fehlerfreies Erlebnis und für die Minimierung der Anwendungslatenz. Die Verfügbarkeit hängt davon ab, ob Ressourcen verfügbar sind und über ausreichend Kapazität verfügen, um den Bedarf zu decken. AWS bietet mehrere Mechanismen zur Verwaltung der Verfügbarkeit. Für Anwendungen, die auf Amazon ECS gehostet werden, gehören dazu Autoscaling und Availability Zones (AZs). Autoscaling verwaltet die Anzahl der Aufgaben oder Instances auf der Grundlage von Metriken, die Sie definieren, während Availability Zones es Ihnen ermöglichen, Ihre Anwendung an isolierten, aber geografisch nahe gelegenen Standorten zu hosten.

Wie bei der Aufgabengröße stellen Kapazität und Verfügbarkeit gewisse Kompromisse dar, die Sie berücksichtigen müssen. Im Idealfall wäre die Kapazität perfekt auf die Nachfrage abgestimmt. Es würde immer gerade genug Kapazität zur Verfügung stehen, um Anfragen zu bearbeiten und Jobs so zu bearbeiten, dass Service Level Objectives (SLOs), einschließlich einer niedrigen Latenz und

Fehlerrate, erfüllt werden. Die Kapazität wäre niemals zu hoch, was zu übermäßigen Kosten führen würde, und sie würde auch niemals zu niedrig sein, was zu hohen Latenzen und Fehlerraten führen würde.

Autoscaling ist ein latenter Prozess. Zunächst müssen Echtzeit-Metriken bereitgestellt werden. CloudWatch Anschließend müssen sie für die Analyse aggregiert werden, was je nach Granularität der Metrik bis zu mehreren Minuten dauern kann. CloudWatch vergleicht die Kennzahlen mit den Alarmschwellenwerten, um einen Mangel oder Überschuss an Ressourcen zu ermitteln. Um Instabilität zu vermeiden, sollten Sie Alarme so konfigurieren, dass der eingestellte Schwellenwert einige Minuten lang überschritten werden muss, bevor der Alarm ausgelöst wird. Außerdem nimmt es Zeit in Anspruch, neue Aufgaben bereitzustellen und Aufgaben zu beenden, die nicht mehr benötigt werden.

Aufgrund dieser potenziellen Verzögerungen im beschriebenen System ist es wichtig, dass Sie durch Überprovisionierung einen gewissen Spielraum wahren. Dies kann dazu beitragen, kurzfristigen Nachfrageschüben Rechnung zu tragen. Auf diese Weise kann Ihre Anwendung auch zusätzliche Anfragen bearbeiten, ohne dass eine Überlastung eintritt. Als bewährte Methode können Sie Ihr Skalierungsziel zwischen 60 und 80% der Auslastung festlegen. Auf diese Weise kann Ihre Anwendung zusätzliche Bedarfsspitzen besser bewältigen, während zusätzliche Kapazität noch bereitgestellt wird.

Ein weiterer Grund, warum wir eine Überversorgung empfehlen, besteht darin, dass Sie schnell auf Ausfälle in der Availability Zone reagieren können. AWS empfiehlt, dass Produktionsworkloads von mehreren Availability Zones aus bedient werden. Dies liegt daran, dass Ihre Aufgaben, die in den verbleibenden Availability Zones ausgeführt werden, auch bei einem Ausfall der Availability Zone den Bedarf decken können. Wenn Ihre Anwendung in zwei Availability Zones ausgeführt wird, müssen Sie die Anzahl Ihrer normalen Aufgaben verdoppeln. Auf diese Weise können Sie bei einem möglichen Ausfall sofort Kapazität bereitstellen. Wenn Ihre Anwendung in drei Availability Zones ausgeführt wird, empfehlen wir, dass Sie das 1,5-fache Ihrer normalen Task-Anzahl ausführen. Das heißt, für jeweils zwei Aufgaben, die für die normale Ausführung erforderlich sind, drei Aufgaben ausführen.

Maximierung der Skalierungsgeschwindigkeit

Autoscaling ist ein reaktiver Prozess, dessen Wirksamkeit einige Zeit in Anspruch nimmt. Es gibt jedoch einige Möglichkeiten, den Zeitaufwand für die Skalierung zu minimieren.

Minimiert die Bildgröße. Das Herunterladen und Entpacken größerer Bilder aus einem Bild-Repository dauert länger. Wenn Sie also die Bildgrößen kleiner halten, verringert sich die Zeit, die ein Container

benötigt, um zu starten. Um die Bildgröße zu reduzieren, können Sie die folgenden spezifischen Empfehlungen befolgen:

- Wenn Sie eine statische Binärdatei erstellen oder Golang verwenden können, erstellen Sie Ihren `FROM` Image-Scratch und fügen Sie nur Ihre Binäranwendung in das resultierende Image ein.
- Verwenden Sie minimierte Basis-Images von Upstream-Distributionsanbietern wie Amazon Linux oder Ubuntu.
- Nehmen Sie keine Build-Artefakte in Ihr endgültiges Image auf. Die Verwendung mehrstufiger Builds kann dabei helfen.
- Kompakte `RUN` Stufen, wo immer dies möglich ist. `RUN`In jeder Phase wird eine neue Bildebene erstellt, was zu einem zusätzlichen Roundtrip zum Herunterladen der Ebene führt. Eine einzelne `RUN` Phase, in der mehrere Befehle miteinander verknüpft sind, && hat weniger Ebenen als eine Phase mit mehreren `RUN` Stufen.
- Wenn Sie Daten, wie z. B. ML-Inferenzdaten, in Ihr endgültiges Bild aufnehmen möchten, schließen Sie nur die Daten ein, die für den Start und die Bereitstellung des Datenverkehrs erforderlich sind. Wenn Sie Daten bei Bedarf von Amazon S3 oder einem anderen Speicher abrufen, ohne den Service zu beeinträchtigen, speichern Sie Ihre Daten stattdessen an diesen Orten.

Bewahren Sie Ihre Bilder in der Nähe auf. Je höher die Netzwerklatenz, desto länger dauert es, das Bild herunterzuladen. Hosten Sie Ihre Bilder in einem Repository in derselben AWS Region, in der sich Ihr Workload befindet. Amazon ECR ist ein leistungsstarkes Image-Repository, das in jeder Region verfügbar ist, in der Amazon ECS verfügbar ist. Vermeiden Sie es, das Internet oder einen VPN-Link zu nutzen, um Container-Images herunterzuladen. Das Hosten Ihrer Bilder in derselben Region verbessert die allgemeine Zuverlässigkeit. Dadurch wird das Risiko von Netzwerkverbindungs- und Verfügbarkeitsproblemen in einer anderen Region gemindert. Alternativ können Sie auch die regionsübergreifende Amazon ECR-Replikation implementieren, um dies zu unterstützen.

Reduzieren Sie die Schwellenwerte für die Integritätsprüfung des Load Balancers. Load Balancer führen Integritätsprüfungen durch, bevor sie Datenverkehr an Ihre Anwendung senden. Die Standardkonfiguration für Integritätsprüfungen für eine Zielgruppe kann 90 Sekunden oder länger dauern. Währenddessen überprüft der Load Balancer den Integritätsstatus und empfängt Anfragen. Wenn Sie das Intervall für Integritätsprüfungen und die Anzahl der Schwellenwerte verringern, kann Ihre Anwendung den Datenverkehr schneller annehmen und die Belastung anderer Aufgaben verringern.

Ziehen Sie die Kaltstartleistung in Betracht. Einige Anwendungen verwenden Laufzeiten, wie z. B. Java, führen Just-In-Time-Kompilierung (JIT) durch. Der Kompilierungsprozess kann zumindest beim Start die Leistung der Anwendung belegen. Eine Behelfslösung besteht darin, die latenzkritischen Teile Ihres Workloads in Sprachen umzuschreiben, die keine Leistungseinbußen beim Kaltstart zur Folge haben.

Verwenden Sie schrittweise Skalierungsrichtlinien, nicht zielgerichtete Skalierungsrichtlinien. Sie haben mehrere Application-Auto-Scaling-Optionen für Amazon-ECS-Aufgaben. Die Zielverfolgung ist der am einfachsten zu verwendende Modus. Damit müssen Sie lediglich einen Zielwert für eine Metrik festlegen, z. B. die durchschnittliche CPU-Auslastung. Anschließend verwaltet der Autoscaler automatisch die Anzahl der Aufgaben, die erforderlich sind, um diesen Wert zu erreichen. Mit der schrittweisen Skalierung können Sie schneller auf Bedarfsänderungen reagieren, da Sie die spezifischen Schwellenwerte für Ihre Skalierungsmetriken definieren und festlegen, wie viele Aufgaben hinzugefügt oder entfernt werden müssen, wenn die Schwellenwerte überschritten werden. Und was noch wichtiger ist: Sie können sehr schnell auf Änderungen der Nachfrage reagieren, indem Sie die Zeitspanne, in der ein Schwellenwertalarm überschritten wird, auf ein Minimum reduzieren. Weitere Informationen finden Sie unter Service Auto Scaling im Amazon Elastic Container Service Developer Guide.

Wenn Sie Amazon EC2 EC2-Instances zur Bereitstellung von Clusterkapazität verwenden, sollten Sie die folgenden Empfehlungen berücksichtigen:

Verwenden Sie größere Amazon EC2 EC2-Instances und schnellere Amazon EBS-Volumes. Sie können die Geschwindigkeit beim Herunterladen und Vorbereiten von Bildern verbessern, indem Sie eine größere Amazon EC2 EC2-Instance und ein schnelleres Amazon EBS-Volume verwenden. Innerhalb einer bestimmten Amazon EC2 EC2-Instance-Familie steigt der maximale Durchsatz von Netzwerk und Amazon EBS mit zunehmender Instance-Größe (z. B. von `m5.xlarge` bis`m5.2xlarge`). Darüber hinaus können Sie Amazon EBS-Volumes anpassen, um deren Durchsatz und IOPS zu erhöhen. Wenn Sie beispielsweise Volumes verwenden, sollten Sie größere `gp2` Volumes verwenden, die einen höheren Basisdurchsatz bieten. Wenn Sie `gp3` Volumes verwenden, geben Sie bei der Erstellung des Volumes den Durchsatz und die IOPS an.

Verwenden Sie den Bridge-Netzwerkmodus für Aufgaben, die auf Amazon EC2 EC2-Instances ausgeführt werden. Aufgaben, die den `bridge` Netzwerkmodus auf Amazon EC2 verwenden, starten schneller als Aufgaben, die den `awsvpc` Netzwerkmodus verwenden. Wenn der `awsvpc` Netzwerkmodus verwendet wird, fügt Amazon ECS der Instance ein elastic network interface (ENI) hinzu, bevor die Aufgabe gestartet wird. Dies führt zu zusätzlicher Latenz. Bei der Verwendung von Bridge-Netzwerken gibt es jedoch mehrere Kompromisse. Für diese Aufgaben gibt es keine eigene

Sicherheitsgruppe, und es gibt einige Auswirkungen auf den Lastenausgleich. Weitere Informationen finden Sie unter Load Balancer-Zielgruppen im Elastic Load Balancing User Guide.

Umgang mit Nachfrageschocks

Bei einigen Anwendungen kommt es zu plötzlichen starken Nachfrageschocks. Dies geschieht aus einer Vielzahl von Gründen: ein Nachrichtenereignis, ein großer Verkauf, ein Medienereignis oder ein anderes Ereignis, das sich viral verbreitet und dazu führt, dass der Traffic in sehr kurzer Zeit schnell und deutlich zunimmt. Ungeplant kann dies dazu führen, dass die Nachfrage die verfügbaren Ressourcen schnell übersteigt.

Der beste Weg, mit Nachfrageschocks umzugehen, besteht darin, sie zu antizipieren und entsprechend zu planen. Da die automatische Skalierung einige Zeit in Anspruch nehmen kann, empfehlen wir, dass Sie Ihre Anwendung skalieren, bevor der Nachfrageschock einsetzt. Um optimale Ergebnisse zu erzielen, empfehlen wir, einen Geschäftsplan zu erstellen, der eine enge Zusammenarbeit zwischen Teams vorsieht, die einen gemeinsamen Kalender verwenden. Das Team, das die Veranstaltung plant, sollte im Voraus eng mit dem für die Bewerbung zuständigen Team zusammenarbeiten. Dies gibt dem Team genügend Zeit, um einen klaren Terminplan zu haben. Sie können die Kapazität so planen, dass sie vor der Veranstaltung skaliert und nach der Veranstaltung wieder erhöht wird. Weitere Informationen finden Sie unter Geplante Skalierung im Benutzerhandbuch für Application Auto Scaling.

Wenn Sie einen Enterprise Support-Plan haben, sollten Sie auch mit Ihrem Technical Account Manager (TAM) zusammenarbeiten. Ihr TAM kann Ihre Servicekontingenten überprüfen und sicherstellen, dass alle erforderlichen Kontingente erhöht werden, bevor die Veranstaltung beginnt. Auf diese Weise erreichen Sie nicht versehentlich irgendwelche Servicekontingenten. Sie können Ihnen auch helfen, indem sie Dienste wie Load Balancer vorwärmen, um sicherzustellen, dass Ihre Veranstaltung reibungslos abläuft.

Der Umgang mit ungeplanten Nachfrageschocks ist ein schwierigeres Problem. Ungeplante Schocks können, sofern sie eine ausreichend große Amplitude haben, schnell dazu führen, dass die Nachfrage die Kapazität übersteigt. Sie kann auch die Reaktionsfähigkeit der automatischen Skalierung übersteigen. Der beste Weg, sich auf ungeplante Schocks vorzubereiten, besteht darin, zu viele Ressourcen bereitzustellen. Sie müssen über genügend Ressourcen verfügen, um den zu erwartenden maximalen Verkehrsbedarf jederzeit bewältigen zu können.

Die Aufrechterhaltung der maximalen Kapazität im Vorgriff auf ungeplante Nachfrageschocks kann kostspielig sein. Um die Auswirkungen auf die Kosten zu minimieren, suchen Sie nach einer

Frühindikatorkennzahl oder einem Ereignis, das darauf hindeutet, dass ein großer Nachfrageschock unmittelbar bevorsteht. Wenn die Kennzahl oder das Ereignis zuverlässig eine deutliche Vorwarnung bietet, beginnen Sie sofort mit dem Skalierungsprozess, wenn das Ereignis eintritt oder wenn die Kennzahl den von Ihnen festgelegten Schwellenwert überschreitet.

Wenn Ihre Anwendung zu plötzlichen, ungeplanten Nachfrageschocks neigt, sollten Sie erwägen, Ihrer Anwendung einen Hochleistungsmodus hinzuzufügen, der unkritische Funktionen einbüßt, aber wichtige Funktionen für einen Kunden beibehält. Gehen Sie beispielsweise davon aus, dass Ihre Anwendung von der Generierung teurer benutzerdefinierter Antworten zur Bereitstellung einer statischen Antwortseite übergehen kann. In diesem Szenario können Sie den Durchsatz erheblich erhöhen, ohne die Anwendung überhaupt skalieren zu müssen.

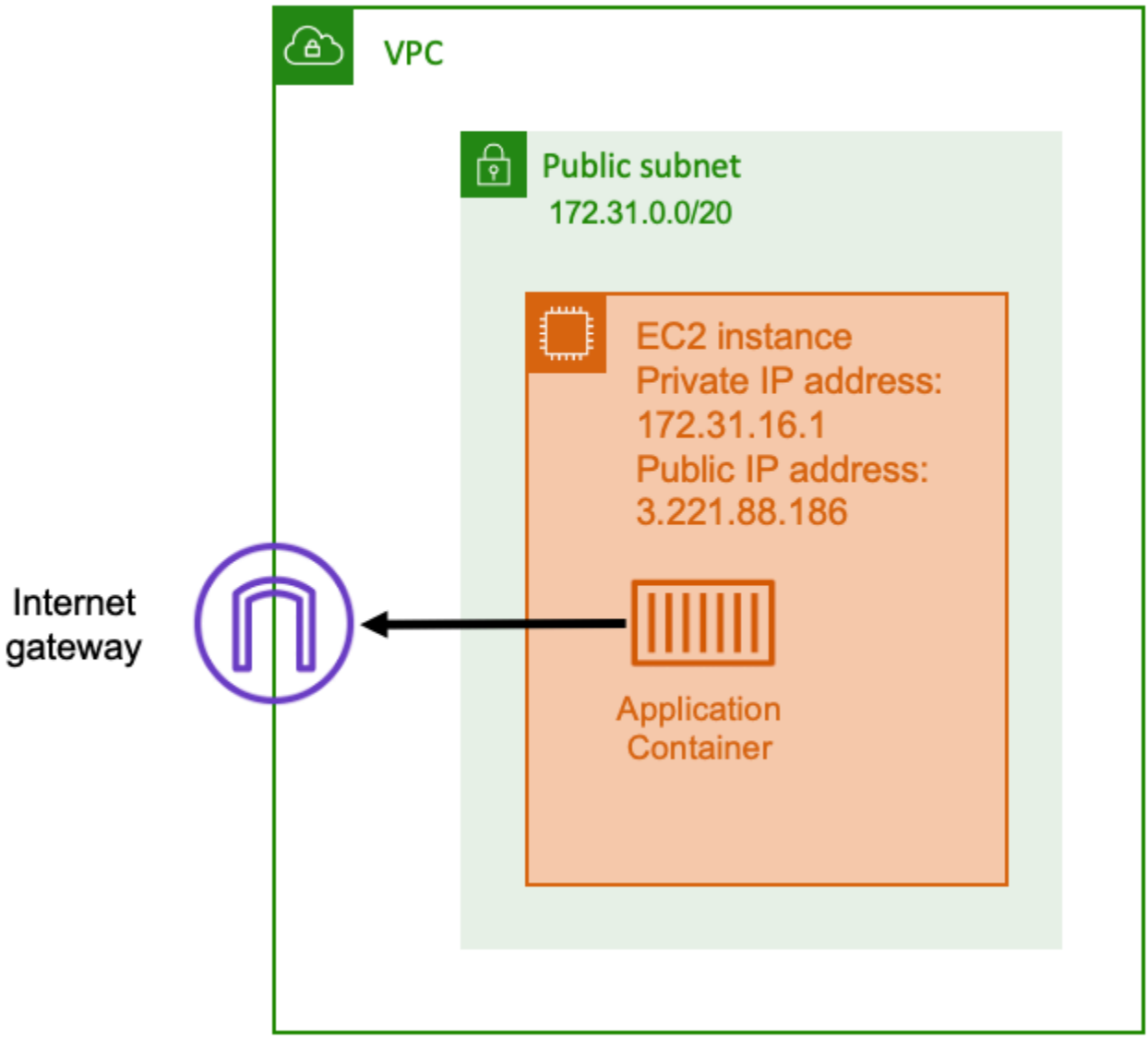
Schließlich können Sie erwägen, monolithische Dienste auseinanderzunehmen, um Nachfrageschocks besser bewältigen zu können. Wenn es sich bei Ihrer Anwendung um einen monolithischen Dienst handelt, der teuer in der Ausführung und langsam zu skalieren ist, können Sie möglicherweise leistungskritische Teile extrahieren oder neu schreiben und sie als separate Dienste ausführen. Diese neuen Dienste können dann unabhängig von weniger kritischen Komponenten skaliert werden. Wenn Sie die Flexibilität haben, leistungskritische Funktionen getrennt von anderen Teilen Ihrer Anwendung zu skalieren, können Sie sowohl den Zeitaufwand für die Kapazitätserweiterung reduzieren als auch Kosten sparen.

Amazon ECS-Anwendungen mit dem Internet Connect

Die meisten containerisierten Anwendungen haben zumindest einige Komponenten, die ausgehenden Zugriff auf das Internet benötigen. Beispielsweise erfordert das Backend für eine mobile App ausgehenden Zugriff auf Push-Benachrichtigungen.

Amazon Virtual Private Cloud bietet zwei Hauptmethoden zur Erleichterung der Kommunikation zwischen Ihrer VPC und dem Internet.

Öffentliches Subnetz und Internet-Gateway



Wenn Sie ein öffentliches Subnetz verwenden, das über eine Route zu einem Internet-Gateway verfügt, kann Ihre containerisierte Anwendung auf einem Host innerhalb einer VPC in einem öffentlichen Subnetz ausgeführt werden. Dem Host, auf dem Ihr Container ausgeführt wird, wird eine öffentliche IP-Adresse zugewiesen. Diese öffentliche IP-Adresse kann vom Internet aus geroutet werden. Weitere Informationen finden Sie unter Internet-Gateways im Amazon-VPC-Benutzerhandbuch.

Diese Netzwerkarchitektur ermöglicht die direkte Kommunikation zwischen dem Host, auf dem Ihre Anwendung ausgeführt wird, und anderen Hosts im Internet. Die Kommunikation ist bidirektional.

Das bedeutet, dass Sie nicht nur eine ausgehende Verbindung zu einem anderen Host im Internet herstellen können, sondern dass auch andere Hosts im Internet versuchen können, eine Verbindung zu Ihrem Host herzustellen. Daher sollten Sie Ihre Sicherheitsgruppen- und Firewallregeln genau beachten. Dadurch wird sichergestellt, dass andere Hosts im Internet keine Verbindungen öffnen können, von denen Sie nicht möchten, dass sie geöffnet werden.

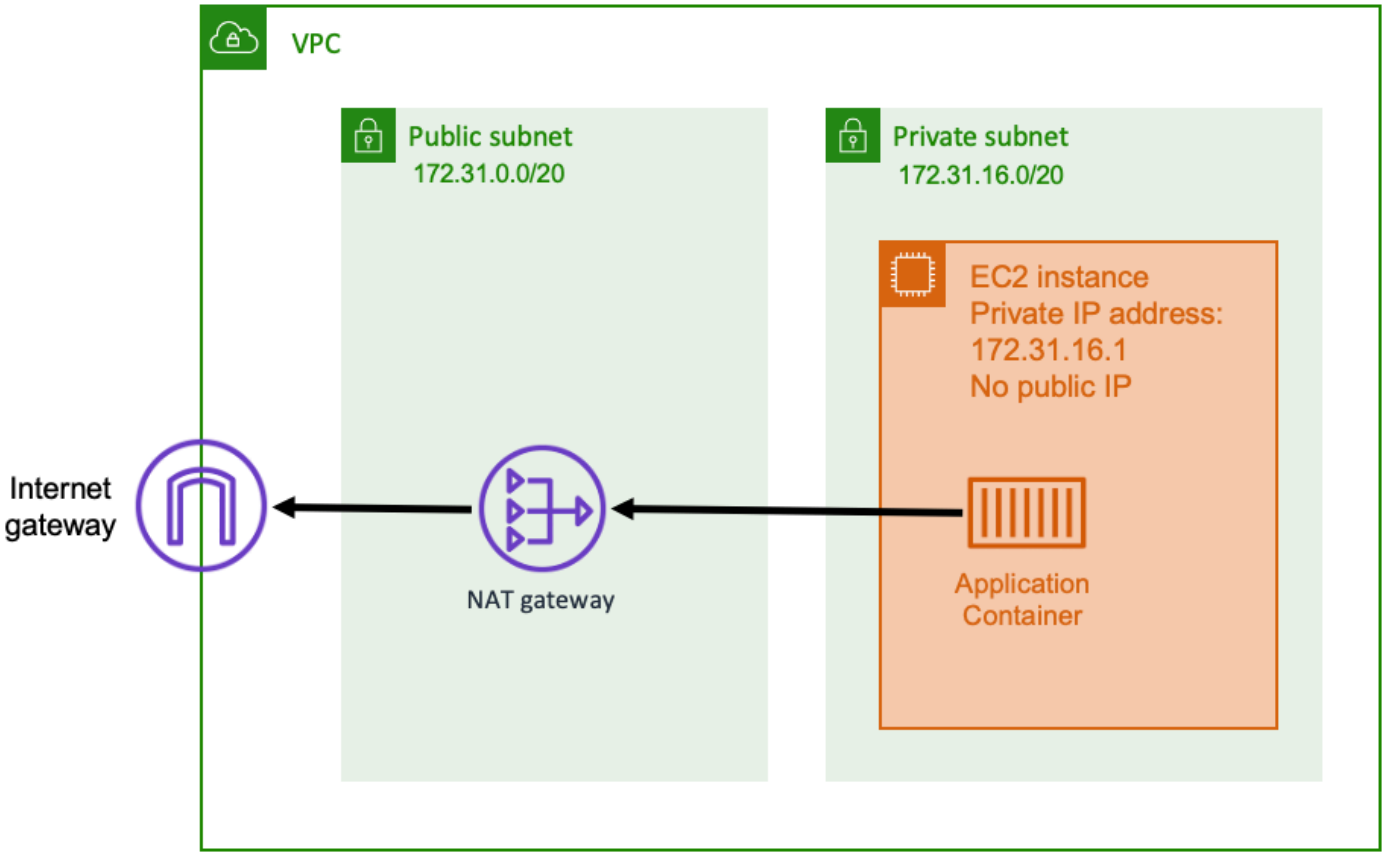
Wenn Ihre Anwendung beispielsweise auf Amazon EC2 läuft, stellen Sie sicher, dass Port 22 für den SSH-Zugriff nicht geöffnet ist. Andernfalls könnte Ihre Instance ständig SSH-Verbindungsversuche von bösartigen Bots im Internet erhalten. Diese Bots durchsuchen öffentliche IP-Adressen. Nachdem sie einen offenen SSH-Port gefunden haben, versuchen sie, mit Brute-Force-Passwörtern auf Ihre Instance zuzugreifen. Aus diesem Grund schränken viele Organisationen die Nutzung öffentlicher Subnetze ein und ziehen es vor, die meisten, wenn nicht sogar alle Ressourcen in privaten Subnetzen zu haben.

Die Verwendung öffentlicher Subnetze für Netzwerke eignet sich für öffentliche Anwendungen, die viel Bandbreite oder minimale Latenz erfordern. Zu den anwendbaren Anwendungsfällen gehören Videostreaming- und Spieledienste.

Dieser Netzwerkansatz wird sowohl bei der Verwendung von Amazon ECS auf Amazon EC2 als auch bei der Verwendung auf AWS Fargate unterstützt.

- Amazon EC2 — Sie können EC2-Instances in einem öffentlichen Subnetz starten. Amazon ECS verwendet diese EC2-Instances als Cluster-Kapazität, und alle Container, die auf den Instances ausgeführt werden, können die zugrunde liegende öffentliche IP-Adresse des Hosts für ausgehende Netzwerke verwenden. Dies gilt sowohl für den als auch für den `host bridge` Netzwerkmodus. Der `awsvpc` Netzwerkmodus stellt jedoch keine öffentlichen IP-Adressen für Task-ENIs bereit. Daher können sie ein Internet-Gateway nicht direkt nutzen.
- Fargate — Wenn Sie Ihren Amazon ECS-Service erstellen, geben Sie öffentliche Subnetze für die Netzwerkkonfiguration Ihres Service an und verwenden Sie die Option Öffentliche IP-Adresse zuweisen. Jede Fargate-Aufgabe ist im öffentlichen Subnetz vernetzt und verfügt über eine eigene öffentliche IP-Adresse für die direkte Kommunikation mit dem Internet.

Privates Subnetz und NAT-Gateway



Wenn Sie ein privates Subnetz und ein NAT-Gateway verwenden, können Sie Ihre containerisierte Anwendung auf einem Host ausführen, der sich in einem privaten Subnetz befindet. Daher hat dieser Host eine private IP-Adresse, die innerhalb Ihrer VPC routbar ist, aber nicht vom Internet aus routbar ist. Das bedeutet, dass andere Hosts innerhalb der VPC über ihre private IP-Adresse eine Verbindung zum Host herstellen können, andere Hosts im Internet jedoch keine eingehende Kommunikation mit dem Host herstellen können.

Bei einem privaten Subnetz können Sie ein NAT-Gateway (Network Address Translation) verwenden, um es einem Host in einem privaten Subnetz zu ermöglichen, eine Verbindung zum Internet herzustellen. Hosts im Internet erhalten eine eingehende Verbindung, die anscheinend von der öffentlichen IP-Adresse des NAT-Gateways stammt, das sich in einem öffentlichen Subnetz befindet. Das NAT-Gateway dient als Brücke zwischen dem Internet und der privaten VPC. Diese Konfiguration wird häufig aus Sicherheitsgründen bevorzugt, da sie bedeutet, dass Ihre VPC vor direktem Zugriff durch Angreifer im Internet geschützt ist. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.

Dieser private Netzwerkansatz eignet sich für Szenarien, in denen Sie Ihre Container vor direktem externen Zugriff schützen möchten. Zu den anwendbaren Szenarien gehören Zahlungsabwicklungssysteme oder Container, in denen Benutzerdaten und Passwörter gespeichert werden. Für das Erstellen und Betreiben eines NAT-Gateways in Ihrem Konto fallen entsprechende Gebühren an. Es gelten auch die stündlichen Nutzungs- und Datenverarbeitungsgebühren für das NAT-Gateway. Aus Redundanzgründen sollten Sie in jeder Availability Zone über ein NAT-Gateway verfügen. Auf diese Weise beeinträchtigt der Verlust der Verfügbarkeit einer einzelnen Availability Zone Ihre ausgehende Konnektivität nicht. Aus diesem Grund kann es bei einer geringen Arbeitslast kostengünstiger sein, private Subnetze und NAT-Gateways zu verwenden.

Dieser Netzwerkansatz wird sowohl bei der Verwendung von Amazon ECS auf Amazon EC2 als auch bei der Verwendung auf AWS Fargate unterstützt.

- Amazon EC2 — Sie können EC2-Instances in einem privaten Subnetz starten. Die Container, die auf diesen EC2-Hosts ausgeführt werden, verwenden das zugrunde liegende Host-Netzwerk, und ausgehende Anfragen werden über das NAT-Gateway abgewickelt.
- Fargate — Wenn Sie Ihren Amazon ECS-Service erstellen, geben Sie private Subnetze für die Netzwerkkonfiguration Ihres Service an und verwenden Sie nicht die Option Öffentliche IP-Adresse zuweisen. Jede Fargate-Aufgabe wird in einem privaten Subnetz gehostet. Der ausgehende Datenverkehr wird über jedes NAT-Gateway geleitet, das Sie mit diesem privaten Subnetz verknüpft haben.

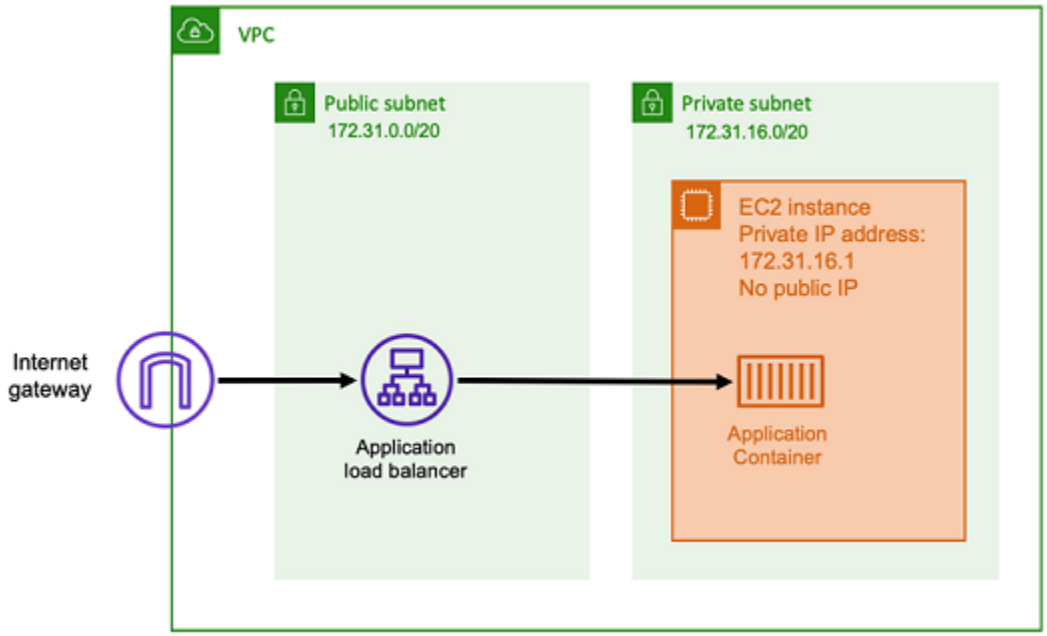
Bewährte Methoden für den Empfang eingehender Verbindungen zu Amazon ECS aus dem Internet

Wenn Sie einen öffentlichen Dienst betreiben, müssen Sie eingehenden Datenverkehr aus dem Internet akzeptieren. Ihre öffentliche Website muss beispielsweise eingehende HTTP-Anfragen von Browsern akzeptieren. In einem solchen Fall müssen auch andere Hosts im Internet eine eingehende Verbindung zum Host Ihrer Anwendung herstellen.

Ein Ansatz zur Lösung dieses Problems besteht darin, Ihre Container auf Hosts zu starten, die sich in einem öffentlichen Subnetz mit einer öffentlichen IP-Adresse befinden. Wir empfehlen dies jedoch nicht für umfangreiche Anwendungen. Für diese ist eine skalierbare Eingabeschicht, die sich zwischen dem Internet und Ihrer Anwendung befindet, ein besserer Ansatz. Für diesen Ansatz können Sie jeden der in diesem Abschnitt aufgeführten AWS Dienste als Eingabe verwenden.

Application Load Balancer

Ein Application Load Balancer funktioniert auf der Anwendungsebene. Es ist die siebte Schicht des Open Systems Interconnection (OSI) -Modells. Dadurch eignet sich ein Application Load Balancer für öffentliche HTTP-Dienste. Wenn Sie über eine Website oder eine HTTP-REST-API verfügen, ist ein Application Load Balancer ein geeigneter Load Balancer für diesen Workload. Weitere Informationen finden Sie unter Was ist ein Application Load Balancer? im Benutzerhandbuch für Application Load Balancers.



Mit dieser Architektur erstellen Sie einen Application Load Balancer in einem öffentlichen Subnetz, sodass er über eine öffentliche IP-Adresse verfügt und eingehende Verbindungen aus dem Internet empfangen kann. Wenn der Application Load Balancer eine eingehende Verbindung, genauer gesagt eine HTTP-Anfrage, empfängt, öffnet er über seine private IP-Adresse eine Verbindung zur Anwendung. Anschließend leitet er die Anfrage über die interne Verbindung weiter.

Ein Application Load Balancer bietet die folgenden Vorteile.

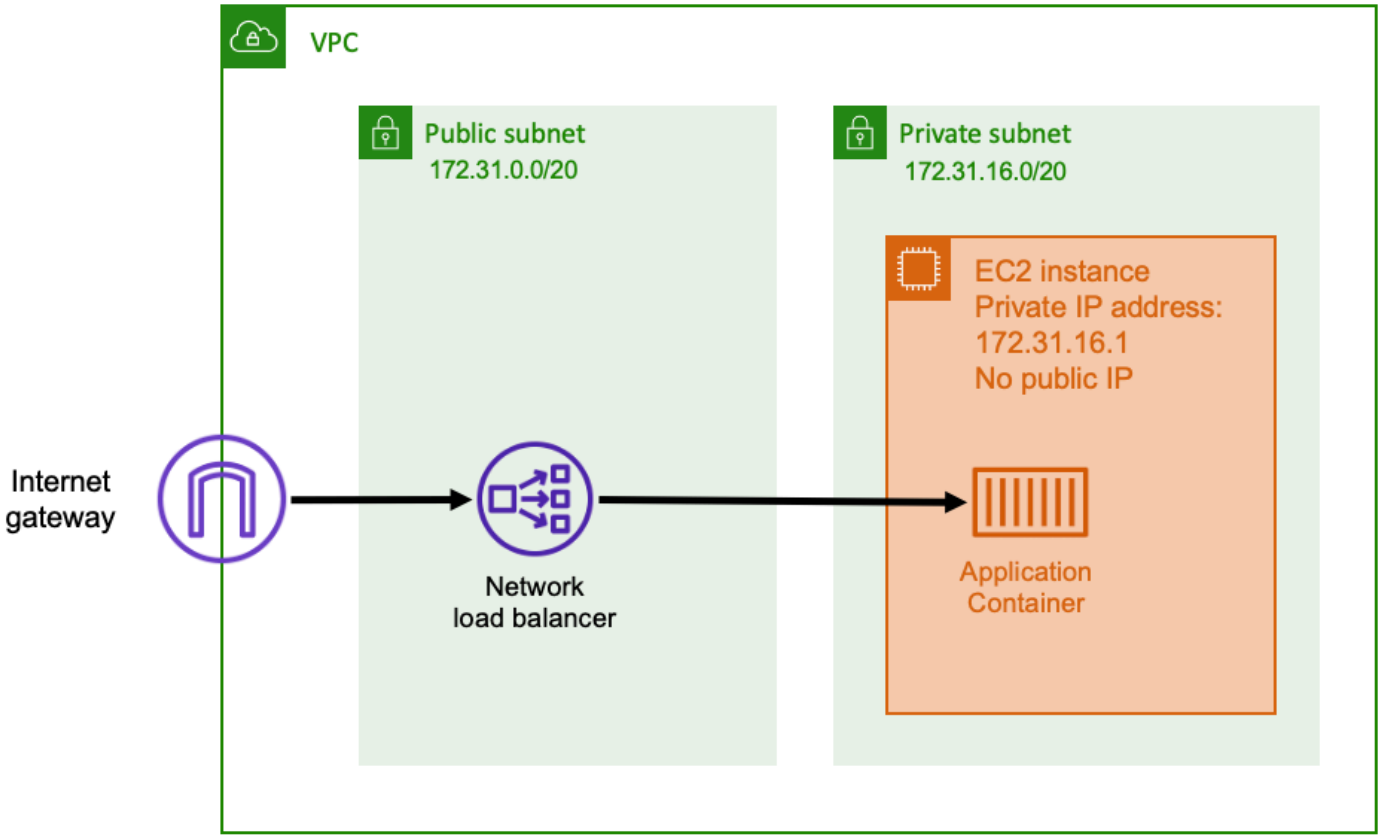
- SSL/TLS-Terminierung — Ein Application Load Balancer kann sichere HTTPS-Kommunikation und Zertifikate für die Kommunikation mit Clients aufrechterhalten. Er kann optional die SSL-Verbindung auf Load Balancer-Ebene beenden, sodass Sie Zertifikate nicht in Ihrer eigenen Anwendung verwalten müssen.
- Erweitertes Routing — Ein Application Load Balancer kann mehrere DNS-Hostnamen haben. Er verfügt außerdem über erweiterte Routing-Funktionen, mit denen eingehende HTTP-Anfragen

anhand von Metriken wie dem Hostnamen oder dem Pfad der Anfrage an verschiedene Ziele gesendet werden können. Das bedeutet, dass Sie einen einzelnen Application Load Balancer als Eingabe für viele verschiedene interne Dienste oder sogar Microservices auf verschiedenen Pfaden einer REST-API verwenden können.

- gRPC-Unterstützung und Websockets — Ein Application Load Balancer kann mehr als nur HTTP verarbeiten. Es kann auch den Lastausgleich von gRPC- und Websocket-basierten Diensten mit HTTP/2-Unterstützung durchführen.
- Sicherheit — Ein Application Load Balancer schützt Ihre Anwendung vor bösartigem Datenverkehr. Er umfasst Funktionen wie Abwehr von HTTP-Synchronisierungen und ist in die AWS Web Application Firewall ()AWS WAF integriert. AWS WAF kann außerdem bösartigen Datenverkehr herausfiltern, der Angriffsmuster wie SQL-Injection oder Cross-Site-Scripting enthalten könnte.

Network Load Balancer

Ein Network Load Balancer arbeitet auf der vierten Ebene der OSI-Modells (Open Systems Interconnection). Es eignet sich für Nicht-HTTP-Protokolle oder Szenarien, in denen end-to-end Verschlüsselung erforderlich ist, verfügt jedoch nicht über dieselben HTTP-spezifischen Funktionen wie ein Application Load Balancer. Daher eignet sich ein Network Load Balancer am besten für Anwendungen, die kein HTTP verwenden. Weitere Informationen finden Sie unter Was ist ein Network Load Balancer? im Benutzerhandbuch für Network Load Balancers.



Wenn ein Network Load Balancer als Eingabe verwendet wird, funktioniert er ähnlich wie ein Application Load Balancer. Das liegt daran, dass er in einem öffentlichen Subnetz erstellt wurde und über eine öffentliche IP-Adresse verfügt, auf die über das Internet zugegriffen werden kann. Der Network Load Balancer öffnet dann eine Verbindung zur privaten IP-Adresse des Hosts, auf dem Ihr Container läuft, und sendet die Pakete von der öffentlichen Seite an die private Seite.

Funktionen des Network Load Balancer

Da der Network Load Balancer auf einer niedrigeren Ebene des Netzwerkstapels arbeitet, verfügt er nicht über dieselben Funktionen wie Application Load Balancer. Er verfügt jedoch über die folgenden wichtigen Funktionen.

- nd-to-end E-Verschlüsselung — Da ein Network Load Balancer auf der vierten Ebene des OSI-Modells arbeitet, liest er den Inhalt von Paketen nicht. Dadurch eignet er sich für den Lastenausgleich von Kommunikationen, die end-to-end verschlüsselt werden müssen.
- TLS-Verschlüsselung — Zusätzlich zur end-to-end Verschlüsselung kann Network Load Balancer auch TLS-Verbindungen beenden. Auf diese Weise müssen Ihre Backend-Anwendungen kein eigenes TLS implementieren.

- UDP-Unterstützung — Da ein Network Load Balancer auf der vierten Ebene des OSI-Modells arbeitet, eignet er sich für Nicht-HTTP-Workloads und andere Protokolle als TCP.

Verbindungen schließen

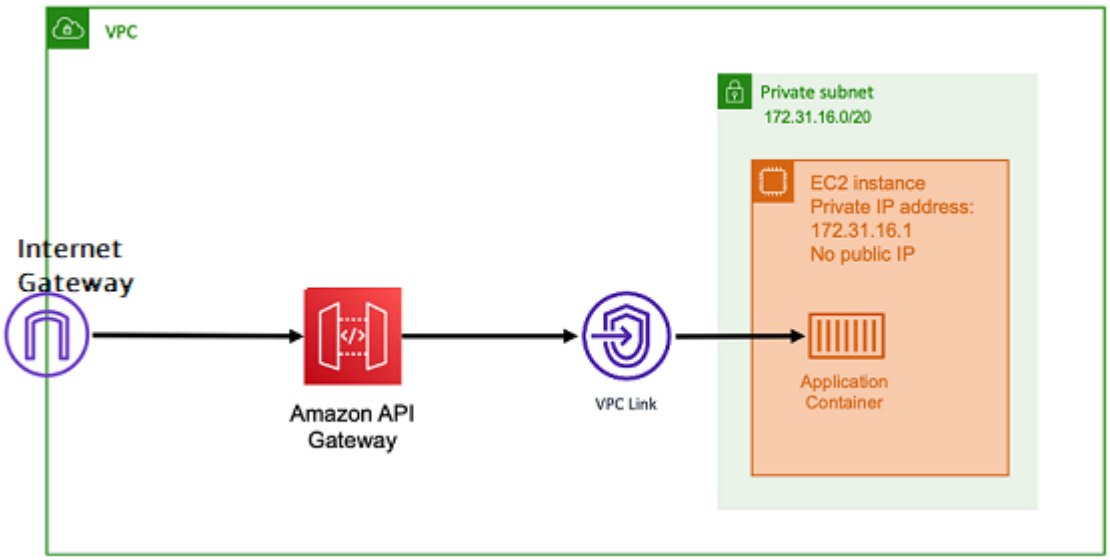
Da der Network Load Balancer das Anwendungsprotokoll auf den höheren Schichten des OSI-Modells nicht beobachtet, kann er in diesen Protokollen keine Abschlussnachrichten an die Clients senden. Im Gegensatz zum Application Load Balancer müssen diese Verbindungen von der Anwendung geschlossen werden. Sie können den Network Load Balancer auch so konfigurieren, dass er die Verbindungen der vierten Ebene schließt, wenn eine Aufgabe gestoppt oder ersetzt wird. Informationen zur Einstellung des Verbindungsabbruchs für Network Load Balancer Balancer-Zielgruppen finden Sie in der Network Load Balancer Balancer-Dokumentation.

Wenn der Network Load Balancer Verbindungen auf der vierten Ebene schließt, kann dies dazu führen, dass Clients unerwünschte Fehlermeldungen anzeigen, wenn der Client sie nicht verarbeitet. Weitere Informationen zur empfohlenen Clientkonfiguration finden Sie hier in der Entwicklerbibliothek.

Die Methoden zum Schließen von Verbindungen variieren je nach Anwendung. Eine Möglichkeit besteht jedoch darin, sicherzustellen, dass die Verzögerung der Zielabmeldung des Network Load Balancer länger ist als das Verbindungstimeout des Clients. Der Client würde zuerst das Timeout überschreiten und über den Network Load Balancer die Verbindung zur nächsten Aufgabe wieder herstellen, während die alte Aufgabe langsam alle Clients leert. Weitere Informationen zur Verzögerung der Network Load Balancer Balancer-Zielabmeldung finden Sie in der Network Load Balancer Balancer-Dokumentation.

Amazon API Gateway API-Gateway-HTTP-API

Amazon API Gateway eignet sich für HTTP-Anwendungen mit plötzlichen oder niedrigen Anforderungsvolumina. Weitere Informationen finden Sie unter Was ist Amazon API Gateway? im API Gateway Developer Guide.



Das Preismodell für Application Load Balancer und Network Load Balancer beinhaltet einen Stundenpreis, damit die Load Balancer jederzeit für die Annahme eingehender Verbindungen verfügbar sind. Im Gegensatz dazu berechnet API Gateway für jede Anfrage separat. Dies hat zur Folge, dass keine Gebühren anfallen, wenn keine Anfragen eingehen. Bei hoher Traffic-Auslastung kann ein Application Load Balancer oder Network Load Balancer ein größeres Anforderungsvolumen zu einem günstigeren Preis pro Anfrage verarbeiten als API Gateway. Wenn Sie jedoch insgesamt nur eine geringe Anzahl von Anfragen oder Zeiten mit geringem Traffic haben, sollte der Gesamtpreis für die Nutzung des API Gateway kostengünstiger sein als die Zahlung einer Stundengebühr für die Wartung eines Load Balancers, der nicht ausgelastet ist. Das API Gateway kann auch API-Antworten zwischenspeichern, was zu niedrigeren Backend-Anforderungsraten führen kann.

API-Gateway-Funktionen, die einen VPC-Link verwenden, der AWS es dem verwalteten Dienst ermöglicht, mithilfe seiner privaten IP-Adresse eine Verbindung zu Hosts innerhalb des privaten Subnetzes Ihrer VPC herzustellen. Es kann diese privaten IP-Adressen anhand von AWS Cloud Map Service Discovery-Datensätzen erkennen, die von Amazon ECS Service Discovery verwaltet werden.

API Gateway unterstützt die folgenden Funktionen.

- Der API-Gateway-Vorgang ähnelt einem Load Balancer, verfügt jedoch über zusätzliche Funktionen, die nur für das API-Management verfügbar sind.
- Das API Gateway bietet zusätzliche Funktionen in Bezug auf Client-Autorisierung, Nutzungsstufen und Änderung von Anforderungen/Antworten. Weitere Informationen finden Sie unter Amazon API Gateway Gateway-Funktionen.

- Das API Gateway kann Edge-API-Gateway-Endpunkte, regionale und private API-Gateway-Endpunkte unterstützen. Edge-Endpunkte sind über eine verwaltete CloudFront Distribution verfügbar. Sowohl regionale als auch private Endpunkte befinden sich lokal in einer Region.
- SSL/TLS-Terminierung
- Routing verschiedener HTTP-Pfade zu verschiedenen Backend-Microservices

Neben den oben genannten Funktionen unterstützt API Gateway auch die Verwendung benutzerdefinierter Lambda-Autorisierer, mit denen Sie Ihre API vor unbefugter Nutzung schützen können. Weitere Informationen finden Sie unter Feldnotizen: Serverlose Container-basierte APIs mit Amazon ECS und Amazon API Gateway.

Greifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu

Um sich von bestimmten Features an- oder abmelden zu können, können Sie Amazon-ECS-Kontoeinstellungen aktivieren oder deaktivieren. Sie können die einzelnen Kontoeinstellungen für jede AWS-Region auf Kontoebene oder für einen bestimmten Benutzer oder eine bestimmte Rolle aktivieren oder deaktivieren.

Möglicherweise möchten Sie sich von bestimmten Features an- oder abmelden, wenn einer der folgenden Punkte für Sie relevant ist:

- Ein Benutzer oder eine Rolle kann bestimmte Kontoeinstellungen für sein individuelles Konto aktivieren oder deaktivieren.
- Ein Benutzer oder eine Rolle kann die Standard-Opt-in- oder Opt-out-Einstellung für alle Benutzer des Kontos festlegen.
- Der Root-Benutzer oder ein Benutzer mit Administratorrechten kann sich für jede bestimmte Rolle oder jeden Benutzer im Konto anmelden oder sich abmelden. Wenn die Kontoeinstellung für den Root-Benutzer geändert wird, wird die Standardeinstellung für alle Benutzer und Rollen festgelegt, für die keine individuelle Kontoeinstellung ausgewählt wurde.

> **Note**
>
> Verbundbenutzer übernehmen die Kontoeinstellung des Root-Benutzers und können keine expliziten Kontoeinstellungen einzeln für sie festlegen.

Die folgenden Kontoeinstellungen sind verfügbar. Sie müssen sich für jede Kontoeinstellung separat an- und abmelden.

Amazon Ressourcennamen (ARNs) und IDs

Ressourcennamen: `serviceLongArnFormat`, `taskLongArnFormat` und `containerInstanceLongArnFormat`

Amazon ECS führt ein neues Format für Amazon-Ressourcennamen (ARNs) und Ressourcen-IDs für Amazon-ECS-Services, -Aufgaben und -Container-Instances ein. Der Opt-In-Status für jeden Ressourcentyp bestimmt das ARN-Format (Amazon Resource Name), das die Ressource verwendet. Sie müssen für das neue ARN-Format einen Opt-in durchführen, um Features wie Ressourcen-Tagging für diesen Ressourcentyp durchführen zu können. Weitere Informationen finden Sie unter Amazon Ressourcennamen (ARNs) und IDs.

Der Standardwert ist `enabled`.

Nur Ressourcen, die nach einem Opt-in gestartet werden, erhalten das neue ARN- und ID-Ressourcen-Format. Für bestehende Ressourcen ändert sich nichts. Damit Amazon-ECS-Services und -Aufgaben auf die neuen ARN- und Ressourcen-ID-Formate umgestellt werden können, müssen Sie den Service oder die Aufgabe neu erstellen. Damit das neue ARN- und Ressourcen-ID-Format für eine Container-Instance verfügbar sind, muss die Container-Instance entfernt und eine neue Container-Instance muss im Cluster gestartet und registriert werden.

> **Note**
>
> Von einem Amazon-ECS-Service gestartete Aufgaben können nur dann das neue ARN- und Ressourcen-ID-Format erhalten, wenn der Service am oder nach dem 16. November 2018 erstellt wurde und der Benutzer, der den Service erstellt hat, das neue Format für Aufgaben gewählt hat.

AWSVPC Bündelung

Ressourcenname: `awsvpcTrunking`

Amazon ECS unterstützt das Starten von Container-Instances mit erhöhter Elastic-Network-Schnittstelle (ENI)-Dichte mithilfe unterstützter Amazon-EC2-Instance-Typen. Wenn Sie diese Instance-Typen verwenden und sich für die neue `awsvpcTrunking`-Kontoeinstellung anmelden, stehen zusätzliche ENIs auf neu gestarteten Container-Instances zur Verfügung.

Sie können diese Konfiguration verwenden, um mehr Aufgaben im Netzwerkmodus `awsvpc` auf jeder Container-Instance zu platzieren. Bei diesem Feature hat eine `c5.large`-Instance mit aktiviertem `awsvpcTrunking` ein ENI-Kontingent von zehn. Die Container-Instance verfügt über eine primäre Netzwerkschnittstelle und Amazon ECS erstellt und fügt eine "Stamm"-Netzwerkschnittstelle an die Container-Instance an. Die primäre Netzwerkschnittstelle und die Trunk-Netzwerkschnittstelle werden dem ENI-Kontingent nicht angerechnet. Daher können Sie diese Konfiguration verwenden, um zehn Aufgaben auf der Container-Instance zu starten, anstatt der derzeitigen zwei Aufgaben. Weitere Informationen finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.

Der Standardwert ist `disabled`.

Nur Ressourcen, die nach einem Opt-in gestartet wurden, erhalten die erhöhten ENI-Limits. Für alle bestehende Ressourcen ändert sich nichts. Damit die erhöhten ENI-Kontingente für eine Container-Instance verfügbar sind, muss die Container-Instance entfernt und eine neue Container-Instance im Cluster registriert werden.

CloudWatch Einblicke in Container

Ressourcenname: `containerInsights`

CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Logs aus Ihren containerisierten Anwendungen und Microservices zusammen. Die Metriken umfassen die Auslastung für Ressourcen wie z B. CPU, Arbeitsspeicher, Datenträger und Netzwerk. Container Insights bietet auch Diagnoseinformationen, wie z. B.Fehler beim Container-Neustart, damit Sie Probleme schnell aufdecken und beheben können. Sie können auch CloudWatch Alarme für Metriken einrichten, die Container Insights sammelt. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.

Wenn Sie für die `containerInsights`-Kontoeinstellung ein Opt-in durchgeführt haben, sind Container Insights standardmäßig für alle neuen Cluster aktiviert. Sie können diese Einstellung für bestimmte Cluster deaktivieren, wenn Sie sie erstellen. Sie können diese Einstellung auch mithilfe der UpdateClusterSettings API ändern.

Für Cluster, die Aufgaben oder Services mit dem Starttyp EC2 enthalten, müssen Ihre Container-Instances Version 1.29.0 oder höher des Amazon-ECS-Agenten ausführen, um Container Insights verwenden zu können. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.

Der Standardwert ist `disabled`.

Dual-Stack-VPC IPv6

Ressourcenname: `dualStackIPv6`

Amazon ECS unterstützt die Bereitstellung von Aufgaben mit einer IPv6-Adresse zusätzlich zur primären privaten IPv4-Adresse.

Damit Aufgaben eine IPv6-Adresse empfangen können, muss die Aufgabe den `awsvpc`-Netzwerkmodus nutzen, muss in einer VPC gestartet werden, die für den Dual-Stack-Modus konfiguriert ist, und die `dualStackIPv6`-Kontoeinstellung muss aktiviert sein. Weitere Informationen zu anderen Anforderungen finden Sie unter Verwenden einer VPC im Dual-Stack-Modus für den EC2-Starttyp und Verwenden einer VPC im Dual-Stack-Modus für den Fargate-Starttyp.

> **Important**
>
> Die `dualStackIPv6`-Kontoeinstellung kann nur mithilfe der Amazon-ECS-API oder AWS CLI geändert werden. Weitere Informationen finden Sie unter Amazon ECS-Kontoeinstellungen ändern.

Wenn Sie eine Aufgabe mit dem `awsvpc`-Netzwerkmodus in einem IPv6-fähigen Subnetz zwischen den Datumsangaben 1. Oktober 2020 und dem 2. November 2020 ausgeführt haben, ist die standardmäßige `dualStackIPv6`-Kontoeinstellung in der Region, in der die Aufgabe ausgeführt wurde, `disabled`. Wenn diese Bedingung nicht erfüllt wird, ist die standardmäßige `dualStackIPv6`-Einstellung in der Region `enabled`.

Der Standardwert ist `disabled`.

FIPS-140-Compliance von Fargate

Ressourcenname: `fargateFIPSMode`

Fargate unterstützt den Bundesstandard für Informationsprozesse (FIPS-140), der die Sicherheitsanforderungen für Verschlüsselungsmodule zum Schutz sensibler Daten festlegt. Es ist der aktuelle Standard der amerikanischen und kanadischen Regierung und gilt für Systeme, die mit dem Federal Information Security Management Act (FISMA) oder dem Federal Risk and Authorization Management Program (FedRAMP) konform sein müssen.

Der Standardwert ist `disabled`.

Sie müssen die FIPS-140-Compliance aktivieren. Weitere Informationen finden Sie unter the section called “AWS Fargate FIPS-140-Konformität”.

> **Important**
>
> Die `fargateFIPSMode`-Kontoeinstellung kann nur mithilfe der Amazon-ECS-API oder AWS CLI geändert werden. Weitere Informationen finden Sie unter Amazon ECS-Kontoeinstellungen ändern.

Ressourcen-Tag-Autorisierung

Ressourcenname: `tagResourceAuthorization`

Einige Amazon-ECS-API-Aktionen erlauben es Ihnen, bei der Erstellung der Ressource Tags anzugeben.

Amazon ECS führt die Tagging-Autorisierung für die Erstellung von Ressourcen ein. Benutzer müssen über Berechtigungen für Aktionen verfügen, mit denen eine Ressource erstellt wird, z. B. `ecsCreateCluster` Wenn bei der Aktion zur Erstellung einer Ressource Tags angegeben wurden, AWS führt dieser Vorgang eine zusätzliche Autorisierung durch, um zu überprüfen, ob Benutzer oder Rollen berechtigt sind, Tags zu erstellen. `ecs:TagResource` Daher müssen Sie explizite Berechtigungen für die Verwendung der Aktion `ecs:TagResource` gewähren. Weitere Informationen finden Sie unter the section called “Zuordnung von Tags (Markierungen) zu Ressourcen während der Erstellung”.

Wartezeit für die Außerbetriebnahme von Fargate-Aufgaben

Ressourcenname: `fargateTaskRetirementWaitPeriod`

AWS ist verantwortlich für das Patchen und die Wartung der zugrunde liegenden Infrastruktur für AWS Fargate. Wenn AWS festgestellt wird, dass für eine auf Fargate gehostete Amazon ECS-Aufgabe ein Sicherheits- oder Infrastruktur-Update erforderlich ist, müssen die Aufgaben gestoppt und neue Aufgaben gestartet werden, um sie zu ersetzen. Sie können die Wartezeit konfigurieren, bis Aufgaben für das Patchen außer Betrieb genommen werden. Sie haben die Möglichkeit, die Aufgabe sofort einzustellen, 7 Kalendertage oder 14 Kalendertage zu warten.

Diese Einstellung befindet sich auf Kontoebene.

Aktivierung von Runtime Monitoring

Ressourcenname: `guardDutyActivate`

Der `guardDutyActivate` Parameter ist in Amazon ECS schreibgeschützt und gibt an, ob Runtime Monitoring von Ihrem Sicherheitsadministrator in Ihrem Amazon ECS-Konto ein- oder ausgeschaltet wurde. GuardDuty steuert diese Kontoeinstellung in Ihrem Namen. Weitere Informationen finden Sie unter Schutz von Amazon ECS-Workloads mit Runtime Monitoring.

Themen

- Amazon Ressourcennamen (ARNs) und IDs
- Zeitachse im ARN- und Ressourcen-ID-Format
- AWS Fargate Einhaltung des Federal Information Processing Standard (FIPS-140)
- Tagging-Autorisierung
- Zeitplan der Tagging-Autorisierung
- AWS Fargate Aufgabe, Stilllegung, Wartezeit
- Laufzeitüberwachung (GuardDuty Amazon-Integration)
- Amazon ECS-Kontoeinstellungen über die Konsole anzeigen
- Amazon ECS-Kontoeinstellungen ändern
- Zurücksetzen auf die Standard-Kontoeinstellungen von Amazon ECS
- Verwaltung der Amazon ECS-Kontoeinstellungen mit dem AWS CLI

Amazon Ressourcennamen (ARNs) und IDs

Wenn Amazon-ECS-Ressourcen erstellt werden, weisen wir jeder Ressource einen eindeutigen Amazon Ressourcennamen (ARN) und eine Ressourcen-ID zu. Wenn Sie ein Befehlszeilen-Tool oder die Amazon-ECS-API für die Arbeit mit Amazon ECS verwenden, sind Ressourcen-ARNs oder -IDs für bestimmte Befehle erforderlich. Wenn Sie beispielsweise den AWS CLI Befehl stop-task verwenden, um eine Aufgabe zu beenden, müssen Sie den ARN oder die ID der Aufgabe im Befehl angeben.

Sie haben die Möglichkeit, das neue Amazon-Ressourcenname (ARN)- und Ressourcen-ID-Formats auf Regionsbasis zu aktivieren oder zu deaktivieren. Aktuell wird jedes neu erstellte Konto standardmäßig aktiviert.

Sie können das Format für den neuen Amazon Ressourcennamen (ARN) und die Ressourcen-ID jederzeit an- oder abmelden. Nachdem Sie sich angemeldet haben, verwenden alle neuen Ressourcen, die Sie erstellen, das neue Format.

> **Note**
>
> Eine Ressourcen-ID ändert sich nicht mehr, nachdem sie erstellt wurde. Daher hat das Opt-in oder Opt-out für das neue Format keinen Einfluss auf Ihre bestehenden Ressourcen-IDs.

In den folgenden Abschnitten wird beschrieben, wie sich die ARN- und Ressourcen-ID-Formate ändern. Weitere Informationen zum Übergang zu den neuen Formaten finden Sie unter Amazon Elastic Container Service – Häufig gestellte Fragen.

Format des Amazon-Ressourcennamens (ARN)

Einige Ressourcen haben einen benutzerfreundlichen Namen (z. B. einen Service namens `production`). In anderen Fällen müssen Sie eine Ressource mit dem Format des Amazon-Ressourcennamens angeben. Das neue ARN-Format für Amazon-ECS-Aufgaben, -Services und -Container-Instances enthält den Namen des Clusters. Weitere Informationen zur Verwendung des neuen ARN-Formats finden Sie unter Amazon ECS-Kontoeinstellungen ändern.

Die folgende Tabelle zeigt sowohl das aktuelle Format als auch das neue Format für jeden Ressourcentyp:

Ressourcentyp	ARN
Container-Instance	Aktuell: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`container-instance-id`* Neu: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`cluster-name`* `/`*`container-instance-id`*
Amazon-ECS-Service	Aktuell: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`service-name`* Neu: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`cluster-name`* `/`*`service-name`*
Amazon-ECS-Aufgabe	Aktuell: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`task-id`* Neu: `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`cluster-name`* `/`*`task-id`*

Länge der Ressourcen-ID

Eine Ressourcen-ID hat die Form einer eindeutigen Kombination von Buchstaben und Zahlen. Neue Ressourcen-ID-Formate enthalten kürzere IDs für Amazon-ECS-Aufgaben und -Container-Instances. Das aktuelle Ressourcen-ID-Format ist 36 Zeichen lang. Die neuen IDs haben 32 Zeichen und enthalten keine Bindestriche. Weitere Informationen zum Verwenden des neuen Ressourcen-ID-Formats finden Sie unter Amazon ECS-Kontoeinstellungen ändern.

Zeitachse im ARN- und Ressourcen-ID-Format

Der Zeitplan für die Opt-in- und Opt-out-Zeiträume für den neuen Amazon-Ressourcennamen (ARN) und das Ressourcen-ID-Format für Amazon-ECS-Ressourcen endete am 1. April 2021. Alle neuen Konten werden standardmäßig für das neue Format aktiviert. Alle neu erstellten Ressourcen erhalten das neue Format, und Sie können es nicht mehr deaktivieren.

AWS Fargate Einhaltung des Federal Information Processing Standard (FIPS-140)

Sie müssen die Einhaltung des Bundesstandards für Informationsprozesse (FIPS-140) auf Fargate aktivieren. Weitere Informationen finden Sie unter the section called “AWS Fargate FIPS-140-Konformität”.

Führen Sie Folgendes aus: `put-account-setting-default` mit der Option `fargateFIPSMode` festgelegt auf `enabled`. Weitere Informationen finden Sie unter put-account-setting-default in der API-Referenz zu Amazon Elastic Container Service.

- Sie können den folgenden Befehl verwenden, um die FIPS-140-Konformität zu aktivieren.

```
aws ecs put-account-setting-default --name fargateFIPSMode --value enabled
```

Beispielausgabe

```
{
    "setting": {
        "name": "fargateFIPSMode",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
         "type": user
    }
```

```
}
```

Sie können `list-account-settings` ausführen, um den aktuellen FIPS-140-Compliancestatus anzuzeigen. Verwenden Sie die Option `effective-settings`, um die Einstellungen auf Kontoebene anzuzeigen.

```
aws ecs list-account-settings --effective-settings
```

Tagging-Autorisierung

Amazon ECS führt die Tagging-Autorisierung für die Erstellung von Ressourcen ein. Benutzer müssen über Tag-Berechtigungen für Aktionen verfügen, mit denen die Ressource erstellt wird, z. B. `ecsCreateCluster` Wenn Sie eine Ressource erstellen und Tags für diese Ressource angeben, AWS führt eine zusätzliche Autorisierung durch, um zu überprüfen, ob Berechtigungen zum Erstellen von Tags vorhanden sind. Daher müssen Sie explizite Berechtigungen für die Verwendung der Aktion `ecs:TagResource` gewähren. Weitere Informationen finden Sie unter the section called “Zuordnung von Tags (Markierungen) zu Ressourcen während der Erstellung”.

Um sich für die Tagging-Autorisierung zu entscheiden, führen Sie `put-account-setting-default` mit der Option `tagResourceAuthorization` aus, wobei diese auf `enable` gesetzt werden muss. Weitere Informationen finden Sie unter put-account-setting-default in der API-Referenz zu Amazon Elastic Container Service. Sie können `list-account-settings` ausführen, um den aktuellen Status der Tagging-Autorisierung einzusehen.

- Sie können den folgenden Befehl verwenden, um die Tagging-Autorisierung zu aktivieren.

```
aws ecs put-account-setting-default --name tagResourceAuthorization --value on --
region region
```

Beispielausgabe

```
{
    "setting": {
        "name": "tagResourceAuthorization",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": user
    }
```

```
}
```

Nachdem Sie die Tagging-Autorisierung aktiviert haben, müssen Sie die entsprechenden Berechtigungen konfigurieren, damit Benutzer Ressourcen bei der Erstellung taggen können. Weitere Informationen finden Sie unter the section called “Zuordnung von Tags (Markierungen) zu Ressourcen während der Erstellung”.

Sie können `list-account-settings` ausführen, um den aktuellen Status der Tagging-Autorisierung einzusehen. Verwenden Sie die Option `effective-settings`, um die Einstellungen auf Kontoebene anzuzeigen.

```
aws ecs list-account-settings --effective-settings
```

Zeitplan der Tagging-Autorisierung

Sie können überprüfen, ob die Tag-Autorisierung aktiv ist, indem Sie den Befehl `list-account-settings` ausführen, um den Wert `tagResourceAuthorization` anzuzeigen. Wenn der Wert on ist, bedeutet dies, dass die Tagging-Autorisierung verwendet wird. Weitere Informationen finden Sie unter list-account-settings in der API-Referenz zu Amazon Elastic Container Service.

Nachfolgend finden Sie die wichtigen Daten im Zusammenhang mit der Tagging-Autorisierung.

- 18. April 2023 – Die Tagging-Autorisierung wird eingeführt. Alle neuen und bestehenden Konten müssen sich für die Nutzung dieses Features entscheiden. Sie können sich dafür entscheiden, ab sofort die Tagging-Autorisierung zu verwenden. Wenn Sie sich für die Nutzung entscheiden, müssen Sie die entsprechenden Berechtigungen gewähren.
- 9. Februar 2024 — 6. März 2024 — Bei allen neuen Konten und bestehenden Konten, die nicht betroffen sind, ist die Tag-Autorisierung standardmäßig aktiviert. Sie können die `tagResourceAuthorization` Kontoeinstellung aktivieren oder deaktivieren, um Ihre IAM-Richtlinie zu überprüfen.

  AWS hat die betroffenen Konten benachrichtigt.

  Um die Funktion zu deaktivieren, müssen Sie `put-account-setting-default` bei der Ausführung die `tagResourceAuthorization` Option auf `off` einstellen.
- 7. März 2024 — Wenn Sie die Tagging-Autorisierung aktiviert haben, können Sie die Kontoeinstellung nicht mehr deaktivieren.

Wir empfehlen Ihnen, Ihre IAM-Richtlinientests vor diesem Datum abzuschließen.

- 29. März 2024 — Alle Konten verwenden die Tagging-Autorisierung. Die Einstellung auf Kontoebene ist in der Amazon ECS-Konsole oder nicht mehr verfügbar. AWS CLI

AWS Fargate Aufgabe, Stilllegung, Wartezeit

AWS sendet Benachrichtigungen, wenn Fargate-Aufgaben auf einer Plattformversion ausgeführt werden, die als veraltet markiert ist. Weitere Informationen finden Sie unter AWS Häufig gestellte Fragen zur Fargate-Aufgabenwartung auf Amazon ECS.

Sie können den Zeitpunkt konfigurieren, zu dem Fargate mit der Außerbetriebnahme der Aufgabe beginnt. Wählen Sie für Workloads, die eine sofortige Anwendung der Updates erfordern, die Einstellung „Sofort“ (`0`). Wenn Sie mehr Kontrolle benötigen, z. B. wenn eine Aufgabe nur während eines bestimmten Zeitfensters gestoppt werden kann, konfigurieren Sie die Optionen 7 Tage (`7`) oder 14 Tage (`14`).

Wir empfehlen Ihnen, eine kürzere Wartezeit zu wählen, damit Sie neuere Versionen der Plattformversionen früher erwerben können.

Konfigurieren Sie die Wartezeit, indem Sie das `put-account-setting-default` Programm ausführen oder `put-account-setting` als Root-Benutzer oder als Administratorbenutzer ausführen. Verwenden Sie die Option `fargateTaskRetirementWaitPeriod` für `name` und die Option `value`, die auf einen der folgenden Werte eingestellt ist:

- `0`- AWS sendet die Benachrichtigung und beginnt sofort, die betroffenen Aufgaben zurückzuziehen.
- `7`- AWS sendet die Benachrichtigung und wartet 7 Kalendertage, bevor mit der Außerbetriebnahme der betroffenen Aufgaben begonnen wird.
- `14` – AWS sendet die Benachrichtigung und wartet 14 Kalendertage, bevor mit der Außerbetriebnahme der betroffenen Aufgaben begonnen wird.

Der Standardwert ist 7 Tage.

Weitere Informationen finden Sie unter put-account-setting-default und put-account-setting in der Referenz zu Amazon Elastic Container Service API.

Sie können den folgenden Befehl ausführen, um die Wartezeit auf 14 Tage festzulegen.

```
aws ecs put-account-setting-default --name fargateTaskRetirementWaitPeriod --value 14
```

Beispielausgabe

```
{
    "setting": {
        "name": "fargateTaskRetirementWaitPeriod",
        "value": "14",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type: user"
    }
}
```

Sie können `list-account-settings` ausführen, um die aktuelle Fargate-Wartezeit für die Außerbetriebnahme der Aufgabe anzuzeigen. Verwenden Sie die Option `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Laufzeitüberwachung (GuardDuty Amazon-Integration)

Runtime Monitoring ist ein intelligenter Dienst zur Bedrohungserkennung, der Workloads schützt, die auf Fargate- und EC2-Container-Instances ausgeführt werden, indem er die AWS Protokoll- und Netzwerkaktivitäten kontinuierlich überwacht, um bösartiges oder unbefugtes Verhalten zu identifizieren.

Der `guardDutyActivate` Parameter ist in Amazon ECS schreibgeschützt und gibt an, ob Runtime Monitoring von Ihrem Sicherheitsadministrator in Ihrem Amazon ECS-Konto ein- oder ausgeschaltet wurde. GuardDuty steuert diese Kontoeinstellung in Ihrem Namen. Weitere Informationen finden Sie unter Schutz von Amazon ECS-Workloads mit Runtime Monitoring.

Sie können ausführen`list-account-settings`, um die aktuelle GuardDuty Integrationseinstellung anzuzeigen.

```
aws ecs list-account-settings
```

Beispielausgabe

```
{
    "setting": {
```

```
        "name": "guardDutyActivate",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:doej",
        "type": aws-managed"
    }
}
```

Amazon ECS-Kontoeinstellungen über die Konsole anzeigen

Sie können den verwenden AWS Management Console , um Ihre Kontoeinstellungen einzusehen.

> **Important**
>
> Die Kontoeinstellungen `dualStackIPv6`, `fargateFIPSMode` und `fargateTaskRetirementWaitPeriod` können nur über AWS CLI angesehen oder geändert werden.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie in der Navigationsleiste oben die Region aus, für die Sie Ihre Kontoeinstellungen anzeigen möchten.
3. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).

Amazon ECS-Kontoeinstellungen ändern

Sie können das verwenden AWS Management Console , um Ihre Kontoeinstellungen zu ändern.

Der `guardDutyActivate` Parameter ist in Amazon ECS schreibgeschützt und gibt an, ob Runtime Monitoring von Ihrem Sicherheitsadministrator in Ihrem Amazon ECS-Konto ein- oder ausgeschaltet wurde. GuardDuty steuert diese Kontoeinstellung in Ihrem Namen. Weitere Informationen finden Sie unter Schutz von Amazon ECS-Workloads mit Runtime Monitoring.

> **Important**
>
> Die Kontoeinstellungen `dualStackIPv6`, `fargateFIPSMode` und `fargateTaskRetirementWaitPeriod` können nur über AWS CLI angesehen oder geändert werden.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie in der Navigationsleiste oben die Region aus, für die Sie Ihre Kontoeinstellungen anzeigen möchten.
3. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
4. Wählen Sie Aktualisieren.
5. Um die Anzahl der Aufgaben, die Sie im awsvpc-Netzwerkmodus für jede EC2-Instance ausführen können, zu erhöhen oder zu verringern, wählen Sie unter AWSVPCTrunking die Option Trunking aus. AWSVPC
6. Um CloudWatch Container Insights standardmäßig für Cluster zu verwenden oder zu beenden, aktivieren oder deaktivieren Sie unter CloudWatch Container Insights die Option Container Insights. CloudWatch
7. Um die Tagging-Autorisierung zu aktivieren oder zu deaktivieren, aktivieren oder deaktivieren Sie unter Resource Tagging Authorization.
8. Wählen Sie Änderungen speichern aus.
9. Wählen Sie auf dem Bestätigungsbildschirm Confirm (Bestätigen) aus, um die Auswahl zu speichern.

Zurücksetzen auf die Standard-Kontoeinstellungen von Amazon ECS

Sie können den verwenden AWS Management Console , um Ihre Amazon ECS-Kontoeinstellungen auf die Standardeinstellungen zurückzusetzen.

Die Option Revert to account default (Auf Standard-Kontoeinstellung zurücksetzen) ist nur verfügbar, wenn Ihre Kontoeinstellungen nicht mehr den Standardeinstellungen entsprechen.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie in der Navigationsleiste oben die Region aus, für die Sie Ihre Kontoeinstellungen anzeigen möchten.
3. Wählen Sie im Navigationsbereich Account Settings (Kontoeinstellungen).
4. Wählen Sie Aktualisieren.
5. Wählen Sie Revert to account default (Auf Standard-Kontoeinstellung zurücksetzen) aus.
6. Wählen Sie auf dem Bestätigungsbildschirm Confirm (Bestätigen) aus, um die Auswahl zu speichern.

Verwaltung der Amazon ECS-Kontoeinstellungen mit dem AWS CLI

Sie können Ihre Kontoeinstellungen mithilfe der Amazon ECS-API AWS CLI oder SDKs verwalten. Die `dualStackIPv6`, `fargateFIPSMode` und die `fargateTaskRetirementWaitPeriod` Kontoeinstellungen können nur mit diesen Tools angezeigt oder geändert werden.

Informationen zu den verfügbaren API-Aktionen für Aufgabendefinitionen finden Sie unter Kontoeinstellungsaktionen in der API-Referenz des Amazon Elastic Container Service.

Verwenden Sie einen der folgenden Befehle, um die Standard-Kontoeinstellungen für alle -Benutzer oder -Rollen in Ihrem Konto zu ändern. Diese Änderungen gelten für das gesamte AWS Konto, sofern ein Benutzer oder eine Rolle diese Einstellungen nicht ausdrücklich für sich selbst außer Kraft setzt.

- put-account-setting-default (AWS CLI)

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled --
region us-east-2
```

  Sie können diesen Befehl auch verwenden, um andere Kontoeinstellungen zu ändern. Hierzu ersetzen Sie den Parameter `name` durch die entsprechende Kontoeinstellung.

- AccountSettingWrite-ECS ()AWS Tools for Windows PowerShell

```
Write-ECSAccountSettingDefault -Name serviceLongArnFormat -Value enabled -Region us-
east-1 -Force
```

So ändern Sie die Kontoeinstellungen für Ihr Benutzerkonto (AWS CLI)

Verwenden Sie einen der folgenden Befehle, um die Kontoeinstellungen für Ihren -Benutzer zu ändern. Wenn Sie diese Befehle als Root-Benutzer ausführen, gelten die Änderungen für das ganze AWS -Konto, es sei den, ein Benutzer oder eine Rolle überschreibt diese Einstellungen explizit für sich selbst.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --region us-
east-1
```

Sie können diesen Befehl auch verwenden, um andere Kontoeinstellungen zu ändern. Hierzu ersetzen Sie den Parameter `name` durch die entsprechende Kontoeinstellung.

- ECS schreiben AccountSetting ()AWS Tools for Windows PowerShell

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -Force
```

So ändern Sie die Kontoeinstellungen für einen bestimmten Benutzer oder eine bestimmte Rolle (AWS CLI)

Verwenden Sie einen der folgenden Befehle und geben Sie den ARN eines Benutzers, einer Rolle oder des Root-Benutzers in der Anfrage an, um die Kontoeinstellungen für einen bestimmten Benutzer oder eine bestimmte Rolle zu ändern.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --principal-
arn arn:aws:iam::aws_account_id:user/principalName --region us-east-1
```

Sie können diesen Befehl auch verwenden, um andere Kontoeinstellungen zu ändern. Hierzu ersetzen Sie den Parameter `name` durch die entsprechende Kontoeinstellung.

- ECS schreiben AccountSetting ()AWS Tools for Windows PowerShell

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -PrincipalArn
 arn:aws:iam::aws_account_id:user/principalName -Region us-east-1 -Force
```

IAM-Rollen für Amazon ECS

Eine IAM-Rolle ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. In Amazon ECS können Sie Rollen erstellen, um Berechtigungen für Amazon ECS-Ressourcen wie Container oder Services zu erteilen.

Die Rollen, die Amazon ECS benötigt, hängen vom Starttyp der Aufgabendefinition und den von Ihnen verwendeten Funktionen ab. Ermitteln Sie anhand der folgenden Tabelle, welche IAM-Rollen Sie für Amazon ECS benötigen.

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Aufgabenausführungsrolle	Diese Rolle ermöglicht es Amazon ECS, andere AWS Dienste in Ihrem Namen zu nutzen.	Ihre Aufgabe wird auf AWS Fargateoder auf externen Instances gehostet und: • ruft ein Container-Image aus einem privaten Amazon ECR-Repository ab. • ruft ein Container-Image aus einem privaten Amazon ECR-Repository in einem anderen Konto als dem Konto ab, das die Aufgabe ausführt. • sendet CloudWatch Container-Protokolle mithilfe des Protokolltreibers an `awslogs` Logs. Ihre Aufgabe wird entweder auf Amazon EC2-Instances AWS Fargate oder auf Amazon EC2 EC2-Instances gehostet und: • verwendet die private Registrie	IAM-Rolle für die Amazon-ECS-Aufgabenausführung

Rolle	Definition	Wenn erforderlich	Weitere Informationen
		rungsauthentifizierung. • verwendet Runtime Monitoring. • Die Aufgabendefinition verweist mithilfe von Secrets Manager Manager-Geheimnissen oder AWS Systems Manager Parameter Store-Parametern auf sensible Daten.	
Aufgabenrolle	Diese Rolle ermöglicht es Ihrem Anwendungscode (auf dem Container), andere AWS Dienste zu verwenden.	Ihre Anwendung greift auf andere AWS Dienste wie Amazon S3 zu.	IAM-Rolle für Amazon ECS-Aufgaben
Rolle für Container-Instances	Diese Rolle ermöglicht es Ihren EC2-Instances oder externen Instances, sich beim Cluster zu registrieren.	Ihre Aufgabe wird auf Amazon EC2 EC2-Instances oder einer externen Instance gehostet.	IAM-Rolle für Amazon-ECS-Container-Instance
Amazon ECS Anywhere Anywhere-Rolle	Diese Rolle ermöglicht Ihren externen Instances den Zugriff auf AWS APIs.	Ihre Aufgabe wird auf externen Instanzen gehostet.	IAM-Rolle in Amazon ECS Anywhere

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Amazon CodeDeploy ECS-Rolle	Diese Rolle ermöglicht CodeDeploy es, Aktualisierungen an Ihren Diensten vorzunehmen.	Sie verwenden den Bereitstellungstyp CodeDeploy Blau/Grün, um Dienste bereitzustellen.	Amazon ECS CodeDeploy IAM-Rolle
Amazon EventBridge ECS-Rolle	Diese Rolle ermöglicht EventBridge es, Aktualisierungen an Ihren Diensten vorzunehmen.	Sie verwenden die EventBridge Regeln und Ziele, um Ihre Aufgaben zu planen.	Amazon ECS EventBridge IAM-Rolle

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Rolle in der Amazon ECS-Infrastruktur	Diese Rolle ermöglicht es Amazon ECS, Infrastrukturressourcen in Ihren Clustern zu verwalten.	• Sie möchten Amazon EBS-Volumes an Ihre Amazon ECS-Aufgaben vom Starttyp Fargate oder EC2 anhängen. Die Infrastrukturrolle ermöglicht es Amazon ECS, Amazon EBS-Volumes für Ihre Aufgaben zu verwalten. • Sie möchten Transport Layer Security (TLS) verwenden, um den Verkehr zwischen Ihren Amazon ECS Service Connect-Services zu verschlüsseln.	IAM-Rolle für die Amazon ECS-Infrastruktur

Amazon-ECS-Aufgabendefinitionen

Eine Aufgabendefinition ist eine Vorlage für Ihre Anwendung. Es handelt sich um eine Textdatei im JSON-Format, die die Parameter und einen oder mehrere Container beschreibt, die Ihre Anwendung bilden.

Die folgenden sind einige der Parameter, die Sie in einer Aufgabendefinition angeben können:

- Der zu verwendende Starttyp, der die Infrastruktur bestimmt, in der Ihre Aufgaben gehostet werden
- Das Docker-Image, das für jeden Container in Ihrer Aufgabe verwendet werden soll.
- Wieviel CPU und Speicher sind für jede Aufgabe oder jeden Container innerhalb einer Aufgabe zu verwenden
- Die Speicher- und CPU-Anforderungen
- Das Betriebssystem des Containers, auf dem die Aufgabe ausgeführt wird
- Der Docker-Netzwerkmodus, der für die Container in Ihrer Aufgabe verwendet werden soll
- Die Protokollierungskonfiguration, die für Ihre Aufgaben verwendet werden soll
- Ob die Aufgabe weiter ausgeführt wird, wenn der Container fertig ist oder fehlschlägt
- Der Befehl, den der Container beim Start ausführt
- Die jeweiligen Daten-Volumes, die mit den Containern in der Aufgabe verwendet werden
- Die IAM-Rolle, die Ihre Aufgaben verwenden

Eine vollständige Liste der Aufgabendefinitionsparameter finden Sie unter Amazon ECS-Aufgabendefinitionsparameter.

Nachdem Sie eine Aufgabendefinition erstellt haben, können Sie die Aufgabendefinition als Aufgabe oder Service ausführen.

- Eine Aufgabe ist die Instance iierung einer Aufgabendefinition auf einer Container-Instance in einem Cluster. Nachdem Sie eine Aufgabendefinition für Ihre Anwendung in Amazon ECS erstellt haben, können Sie angeben, wie viele Aufgaben in Ihrem Cluster ausgeführt werden.
- Ein Amazon-ECS-Service führt Ihre gewünschte Anzahl von Aufgaben gleichzeitig in einem Amazon-ECS-Cluster aus und wartet sie. Wenn eine Ihrer Aufgaben aus irgendeinem Grund fehlschlägt oder angehalten wird, launcht der Scheduler des Amazon-ECS-Service eine andere

Instance entsprechend Ihrer Aufgabendefinition. Dies geschieht, um sie zu ersetzen und dadurch Ihre gewünschte Anzahl von Aufgaben im Service beizubehalten.

Themen

Status der Amazon ECS-Aufgabendefinition

Der Status einer Aufgabendefinition ändert sich, wenn Sie sie erstellen, abmelden oder löschen. Sie können den Status der Aufgabendefinition in der Konsole oder mithilfe von anzeigen. `DescribeTaskDefinition`

Die folgenden Zustände sind für eine Aufgabendefinition möglich:

ACTIVE

Eine Aufgabendefinition ist `ACTIVE`, nachdem sie bei Amazon ECS registriert wurde. Sie können Aufgabendefinitionen im `ACTIVE`-Zustand verwenden, um Aufgaben auszuführen oder Services zu erstellen.

INACTIVE

Eine Aufgabendefinition geht vom `ACTIVE`-Status in den `INACTIVE`-Status über, wenn Sie die Registrierung einer Aufgabendefinition aufheben. Sie können eine `INACTIVE`-Aufgabendefinition abrufen, indem Sie `DescribeTaskDefinition` aufrufen. Sie können keine neuen Aufgaben ausführen oder neue Services erstellen, wenn sich eine Aufgabendefinition in dem `INACTIVE`-Status befindet. Es gibt keine Auswirkungen auf bestehende Services oder Aufgaben.

DELETE_IN_PROGRESS

Eine Aufgabendefinition geht vom `INACTIVE`-Status in den `DELETE_IN_PROGRESS`-Status über, nachdem Sie die Aufgabendefinition zum Löschen eingereicht haben. Nachdem die Aufgabendefinition den `DELETE_IN_PROGRESS`-Status erreicht hat, überprüft Amazon ECS in regelmäßigen Abständen, ob die Ziel-Aufgabendefinition von keinen aktiven Aufgaben oder Bereitstellungen referenziert wird, und löscht die Aufgabendefinition anschließend dauerhaft. Sie können keine neuen Aufgaben ausführen oder neue Services erstellen, wenn sich eine Aufgabendefinition in dem `DELETE_IN_PROGRESS`-Status befindet. Eine Aufgabendefinition kann jederzeit zum Löschen eingereicht werden, ohne dass sich dies auf bestehende Aufgaben und Dienste auswirkt.

Aufgabendefinitionen, die sich im `DELETE_IN_PROGRESS`-Status befinden, können in der Konsole angezeigt werden, und Sie können die Aufgabendefinition aufrufen, indem Sie `DescribeTaskDefinition` aufrufen.

Wenn Sie alle Versionen der `INACTIVE`-Aufgabendefinition löschen, wird der Name der Aufgabendefinition nicht in der Konsole angezeigt und nicht in der API zurückgegeben. Wenn sich eine Version der Aufgabendefinition im `DELETE_IN_PROGRESS` Status befindet, wird der Name der Aufgabendefinition in der Konsole angezeigt und in der API zurückgegeben. Der Name der Aufgabendefinition wird von Amazon ECS beibehalten und die Version wird erhöht, wenn Sie das nächste Mal eine Aufgabendefinition mit diesem Namen erstellen.

Wenn Sie AWS Config Ihre Aufgabendefinitionen verwalten, werden Ihnen alle Registrierungen von Aufgabendefinitionen in AWS Config Rechnung gestellt. Ihnen wird nur die Abmeldung der letzten `ACTIVE`-Aufgabendefinition in Rechnung gestellt. Das Löschen einer Aufgabendefinition ist kostenlos. Weitere Informationen über die Preise finden Sie unter AWS Config – Preise.

Amazon-ECS-Ressourcen, die einen Löschvorgang blockieren können

Eine Anforderung zum Löschen von Aufgabendefinitionen wird nicht abgeschlossen, wenn Amazon ECS-Ressourcen vorhanden sind, die von der Revision der Aufgabendefinition abhängen. Die folgenden Ressourcen können verhindern, dass eine Aufgabendefinition gelöscht wird:

- Amazon-ECS-Aufgaben – Die Aufgabendefinition ist erforderlich, damit die Aufgabe fehlerfrei bleibt.

- Amazon-ECS-Bereitstellungen und Aufgabensätze – Die Aufgabendefinition ist erforderlich, wenn ein Skalierungsereignis für eine Amazon-ECS-Bereitstellung oder einen Aufgabensatz ausgelöst wird.

Bleibt Ihre Aufgabendefinition unverändert, `DELETE_IN_PROGRESS` können Sie die Konsole oder die verwenden, AWS CLI um die Ressourcen zu identifizieren und dann zu beenden, die das Löschen der Aufgabendefinition blockieren.

Löschen der Aufgabendefinition, nachdem die blockierende Ressource entfernt wurde

Die folgenden Regeln gelten, nachdem Sie die Ressourcen entfernt haben, die das Löschen der Aufgabendefinition blockieren:

- Amazon-ECS-Aufgaben – Nach dem Beenden der Aufgabe kann es bis zu 1 Stunde dauern, bis das Löschen der Aufgabendefinition abgeschlossen ist.
- Amazon-ECS-Bereitstellungen und Aufgabensätze – Es kann bis zu 24 Stunden dauern, bis das Löschen der Aufgabendefinition abgeschlossen ist, nachdem die Bereitstellung oder der Aufgabensatz gelöscht wurde.

Entwickeln Sie Ihre Anwendung für Amazon ECS

Sie entwerfen Ihre Anwendung, indem Sie eine Aufgabendefinition für Ihre Anwendung erstellen. Die Aufgabendefinition enthält die Parameter, die Informationen über die Anwendung definieren, darunter:

- Der zu verwendende Starttyp, der die Infrastruktur bestimmt, auf der Ihre Aufgaben gehostet werden.

  Wenn Sie den EC2-Starttyp verwenden, wählen Sie auch den Instance-Typ aus. Für einige Instance-Typen, z. B. GPU, müssen Sie zusätzliche Parameter festlegen. Weitere Informationen finden Sie unter Anwendungsfälle für Amazon ECS-Aufgabendefinitionen.
- Das Container-Image, das Ihren Anwendungscode und alle Abhängigkeiten enthält, die Ihr Anwendungscode zur Ausführung benötigt.
- Der Netzwerkmodus, der für die Container in Ihrer Aufgabe verwendet werden soll

  Der Netzwerkmodus bestimmt, wie Ihre Aufgabe über das Netzwerk kommuniziert.

Für Aufgaben, die auf einer EC2-Instance ausgeführt werden, gibt es mehrere Optionen, wir empfehlen jedoch, den `awsvpc` Netzwerkmodus zu verwenden. Der `awsvpc` Netzwerkmodus vereinfacht Container-Netzwerke, da Sie mehr Kontrolle darüber haben, wie Ihre Anwendungen miteinander und mit anderen Diensten innerhalb Ihrer VPCs kommunizieren.

Für Aufgaben, die auf Fargate ausgeführt werden, können Sie nur den `awsvpc` Netzwerkmodus verwenden.

- Die für Ihre Aufgaben zu verwendende Protokollierungskonfiguration.
- Alle Datenvolumes, die mit den Containern in der Aufgabe verwendet werden.

Eine vollständige Liste der Aufgabendefinitionsparameter finden Sie unter Amazon ECS-Aufgabendefinitionsparameter.

Beachten Sie bei der Erstellung Ihrer Aufgabendefinitionen die folgenden Richtlinien:

- Verwenden Sie jede Aufgabendefinitionsfamilie nur für einen Geschäftszweck.

  Wenn Sie mehrere Typen von Anwendungscontainern in derselben Aufgabendefinition zusammenfassen, können Sie diese Container nicht unabhängig voneinander skalieren. Es ist beispielsweise unwahrscheinlich, dass sowohl eine Website als auch eine API mit derselben Geschwindigkeit aufskaliert werden müssen. Mit steigendem Datenverkehr wird eine andere Anzahl von Web-Containern benötigt als API-Container. Wenn diese beiden Container in derselben Aufgabendefinition bereitgestellt werden, führt jede Aufgabe dieselbe Anzahl von Webcontainern und API-Containern aus.
- Ordnen Sie jeder Anwendungsversion eine Revision der Aufgabendefinition innerhalb einer Aufgabendefinitionsfamilie zu.

  Betrachten Sie innerhalb einer Aufgabendefinitionsfamilie jede Version der Aufgabendefinition als eine Momentaufnahme der Einstellungen für ein bestimmtes Container-Image. Dies ist vergleichbar mit der Art und Weise, wie der Container eine Momentaufnahme aller Dinge ist, die zur Ausführung einer bestimmten Version Ihres Anwendungscodes erforderlich sind.

  Stellen Sie sicher, dass eine one-to-one Zuordnung zwischen einer Version des Anwendungscodes, einem Container-Image-Tag und einer Revision der Aufgabendefinition besteht. Ein typischer Release-Prozess beinhaltet einen Git-Commit, der in ein Container-Image umgewandelt wird, das mit dem Git-Commit-SHA gekennzeichnet ist. Dann erhält dieses Container-Image-Tag seine eigene Version der Amazon-ECS-Aufgabendefinition. Zuletzt

wird der Amazon-ECS-Service aktualisiert, um ihm mitzuteilen, dass er die neue Version der Aufgabendefinition bereitstellen soll.

- Verwenden Sie für jede Aufgabendefinitionsfamilie unterschiedliche IAM-Rollen.

  Definieren Sie jede Aufgabendefinition mit einer eigenen IAM-Rolle. Diese Empfehlung sollte mit unserer Empfehlung einhergehen, jeder Geschäftskomponente eine eigene Aufgabendefinitionsfamilie zur Verfügung zu stellen. Durch die Implementierung dieser beiden bewährten Methoden können Sie einschränken, wie viel Zugriff jeder Dienst auf Ressourcen in Ihrem AWS Konto hat. Sie können Ihrem Authentifizierungsservice beispielsweise Zugriff gewähren, um eine Verbindung zu Ihrer Kennwortdatenbank herzustellen. Gleichzeitig können Sie auch sicherstellen, dass nur Ihr Bestellservice Zugriff auf die Kreditkartenzahlungsinformationen hat.

Bewährte Methoden für Amazon ECS-Container-Images

Ein Container-Image besteht aus einer Reihe von Anweisungen zum Erstellen des Containers. Ein Container-Image enthält Ihren Anwendungscode und alle Abhängigkeiten, die Ihr Anwendungscode zur Ausführung benötigt. Zu den Anwendungsabhängigkeiten gehören die Quellcodepakete, auf denen Ihr Anwendungscode basiert sowie eine Sprachlaufzeit für interpretierte Sprachen und Binärpakete, auf denen Ihr dynamisch verlinkter Code basiert.

Beachten Sie beim Entwerfen und Erstellen Ihrer Container-Images die folgenden Richtlinien:

- Vervollständigen Sie Ihre Container-Images, indem Sie alle Anwendungsabhängigkeiten als statische Dateien im Container-Image speichern.

  Wenn Sie etwas am Container-Image ändern, erstellen Sie mit den Änderungen ein neues Container-Image.
- Führen Sie einen einzelnen Anwendungsprozess in einem Container aus.

  Die Lebensdauer des Containers ist so lang, wie der Anwendungsprozess läuft. Amazon ECS ersetzt abgestürzte Prozesse und bestimmt, wo der Ersatzprozess gestartet werden soll. Ein vollständiges Image macht die gesamte Bereitstellung robuster.
- Sorgen Sie dafür, dass Ihre Anwendung verwaltet wird`SIGTERM`.

  Wenn Amazon ECS eine Aufgabe stoppt, sendet es zunächst ein SIGTERM-Signal an die Aufgabe, um die Anwendung darüber zu informieren, dass sie beendet und heruntergefahren werden muss. Amazon ECS sendet dann eine `SIGKILL` Nachricht. Wenn Anwendungen das

ignorierenSIGTERM, muss der Amazon ECS-Service warten, bis das SIGKILL Signal gesendet wird, um den Prozess zu beenden.

Sie müssen ermitteln, wie lange es dauert, bis Ihre Anwendung ihre Arbeit abgeschlossen hat, und sicherstellen, dass Ihre Anwendungen das SIGTERM Signal verarbeiten. Die Signalverarbeitung der Anwendung muss verhindern, dass die Anwendung neue Aufgaben annimmt und die laufende Arbeit abschließt, oder unerledigte Arbeiten außerhalb der Aufgabe speichern, wenn die Arbeit zu lange dauert.

- Konfigurieren Sie containerisierte Anwendungen so, dass sie Protokolle in stdout und stderr schreiben.

  Durch die Entkopplung der Protokollverarbeitung von Ihrem Anwendungscode können Sie die Protokollverarbeitung flexibel auf Infrastrukturebene anpassen. Ein Beispiel hierfür ist die Änderung Ihres Protokollierungssystems. Anstatt Ihre Dienste zu ändern und ein neues Container-Image zu erstellen und bereitzustellen, können Sie die Einstellungen anpassen.

- Verwenden Sie Tags, um Ihre Container-Images zu versionieren.

  Container-Images werden in einer Container-Registrierung gespeichert. Jedes Image in einer Registrierung wird durch ein Tag identifiziert. Es gibt ein Tag namens latest. Dieses Tag dient als Hinweis auf die neueste Version des Anwendungs-Container-Images, ähnlich wie HEAD in einem Git-Repository. Wir empfehlen, dass Sie das latest-Tag nur für Tests verwenden. Es hat sich bewährt, Container-Images mit einem eindeutigen Tag für jeden Build zu kennzeichnen. Wir empfehlen, dass Sie Ihre Images mit dem git-SHA für den Git-Commit taggen, der zum Erstellen des Images verwendet wurde.

  Sie müssen nicht für jeden Commit ein Container-Image erstellen. Wir empfehlen jedoch, dass Sie jedes Mal, wenn Sie einen bestimmten Code-Commit für die Produktionsumgebung veröffentlichen, ein neues Container-Image erstellen. Wir empfehlen außerdem, das Image mit einem Tag zu versehen, das dem git-Commit des Codes entspricht, der sich im Image befindet. Wenn Sie das Image mit dem Tag des git-Commits versehen, können Sie schneller herausfinden, welche Version des Codes auf dem Image ausgeführt wird.

  Wir empfehlen außerdem, dass Sie unveränderliche Image-Tags in der Amazon Elastic Container Registry aktivieren. Mit dieser Einstellung können Sie das Container-Image, auf das ein Tag verweist, nicht ändern. Stattdessen erzwingt Amazon ECR, dass ein neues Bild in ein neues Tag hochgeladen werden muss. Weitere Informationen finden Sie unter Veränderlichkeit von Image-Tags im Amazon-ECR-Benutzerhandbuch.

Wenn Sie Ihre Anwendung für die Ausführung entwerfen AWS Fargate, müssen Sie entscheiden, ob Sie mehrere Container in derselben Aufgabendefinition bereitstellen oder Container separat in mehreren Aufgabendefinitionen bereitstellen möchten. Wenn die folgenden Bedingungen erforderlich sind, wird empfohlen, mehrere Container in derselben Aufgabendefinition bereitzustellen:

- Ihre Container haben einen gemeinsamen Lebenszyklus (d. h. sie werden gemeinsam gestartet und beendet).
- Ihre Container müssen auf demselben zugrunde liegenden Host ausgeführt werden (d. h. ein Container verweist auf den anderen auf einem localhost-Port).
- Ihre Container teilen Ressourcen.
- Ihre Container nutzen Daten-Volumes gemeinsam.

Wenn diese Bedingungen nicht erforderlich sind, empfehlen wir, Container separat in mehreren Aufgabendefinitionen bereitzustellen. Auf diese Weise können Sie die Container separat skalieren, bereitstellen und deprovisionieren.

Bewährte Methoden für Amazon ECS-Aufgabengrößen

Eine der wichtigsten Entscheidungen, die Sie bei der Bereitstellung von Containern auf Amazon ECS treffen müssen, ist Ihre Container- und Aufgabengröße. Sowohl Ihre Container- als auch Ihre Aufgabengröße sind für die Skalierung und Kapazitätsplanung von entscheidender Bedeutung. In Amazon ECS gibt es zwei Ressourcenmetriken, die für die Kapazität verwendet werden: CPU und Arbeitsspeicher. Die CPU wird in Einheiten von 1/1024 einer vollen vCPU gemessen (wobei 1024 Einheiten einer ganzen vCPU entsprechen). Der Arbeitsspeicher wird in Megabyte gemessen. In Ihrer Aufgabendefinition können Sie Ressourcenreservierungen und -limits deklarieren.

Wenn Sie eine Reservierung deklarieren, geben Sie die Mindestmenge an Ressourcen an, die eine Aufgabe benötigt. Ihre Aufgabe erhält mindestens die angeforderte Menge an Ressourcen. Ihre Anwendung kann möglicherweise mehr CPU oder Arbeitsspeicher verwenden als die Reservierung, die Sie deklariert haben. Dies unterliegt jedoch allen Beschränkungen, die Sie ebenfalls deklariert haben. Wenn Sie mehr als den Reservierungsbetrag verwenden, wird dies als Bursting bezeichnet. In Amazon ECS sind Reservierungen garantiert. Wenn Sie beispielsweise Amazon EC2 EC2-Instances verwenden, um Kapazität bereitzustellen, platziert Amazon ECS keine Aufgabe auf einer Instance, bei der die Reservierung nicht erfüllt werden kann.

Ein Limit ist die maximale Menge an CPU-Einheiten oder Arbeitsspeicher, die Ihr Container oder Ihre Aufgabe verwenden kann. Jeder Versuch, mehr CPUs als diesen Grenzwert zu verwenden, führt

zu einer Drosselung. Jeder Versuch, mehr Speicher zu verwenden, führt dazu, dass Ihr Container gestoppt wird.

Die Auswahl dieser Werte kann eine Herausforderung sein. Das liegt daran, dass die Werte, die für Ihre Anwendung am besten geeignet sind, stark von den Ressourcenanforderungen Ihrer Anwendung abhängen. Auslastungstests Ihrer Anwendung sind der Schlüssel zu einer erfolgreichen Planung des Ressourcenbedarfs und zu einem besseren Verständnis der Anforderungen Ihrer Anwendung.

Zustandslose Anwendungen

Für statusfreie Anwendungen, die horizontal skaliert werden, wie z. B. eine Anwendung hinter einem Load Balancer, empfehlen wir, dass Sie zunächst ermitteln, wie viel Speicher Ihre Anwendung bei der Bearbeitung von Anfragen verbraucht. Zu diesem Zweck können Sie herkömmliche Tools wie `ps` oder oder oder `top` Überwachungslösungen wie CloudWatch Container Insights verwenden.

Denken Sie bei der Festlegung einer CPU-Reservierung darüber nach, wie Sie Ihre Anwendung skalieren möchten, um Ihren Geschäftsanforderungen gerecht zu werden. Sie können kleinere CPU-Reservierungen, wie z. B. 256 CPU-Einheiten (oder 1/4 vCPU), verwenden, um feinkörnig zu skalieren und so die Kosten zu minimieren. Sie können jedoch möglicherweise nicht schnell genug skaliert werden, um erhebliche Nachfragespitzen zu bewältigen. Sie können größere CPU-Reservierungen verwenden, um schneller ein- und auszuskalieren und so Nachfragespitzen schneller zu begegnen. Größere CPU-Reservierungen sind jedoch teurer.

Andere Anwendungen

Bei Anwendungen, die nicht horizontal skaliert werden können, wie z. B. Singleton Worker oder Datenbankserver, stellen die verfügbaren Kapazitäten und Kosten die wichtigsten Überlegungen dar. Sie sollten die Größe des Speichers und der CPU auf der Grundlage der Belastungstests auswählen, dass Sie Datenverkehr bereitstellen müssen, um Ihr Service-Level-Ziel zu erreichen. Amazon ECS stellt sicher, dass die Anwendung auf einem Host mit ausreichender Kapazität platziert wird.

Bewährte Methoden zur Netzwerksicherheit für Amazon ECS

Netzwerksicherheit ist ein breit gefächertes Thema, das mehrere Unterthemen umfasst. Dazu gehören encryption-in-transit Netzwerksegmentierung und -isolierung, Firewalling, Datenverkehrs-Routing und Beobachtbarkeit.

Verschlüsselung während der Übertragung

Die Verschlüsselung des Netzwerkverkehrs verhindert, dass unbefugte Benutzer Daten abfangen und lesen können, wenn diese Daten über ein Netzwerk übertragen werden. Mit Amazon ECS kann die Netzwerkverschlüsselung auf eine der folgenden Arten implementiert werden.

- Mit einem Service Mesh (TLS):

  Mit AWS App Mesh können Sie TLS-Verbindungen zwischen den Envoy-Proxys konfigurieren, die mit Mesh-Endpunkten bereitgestellt werden. Zwei Beispiele sind virtuelle Knoten und virtuelle Gateways. Die TLS-Zertifikate können von (ACM) stammen AWS Certificate Manager . Sie können auch von Ihrer eigenen privaten Zertifizierungsstelle stammen.

  - Transport Layer Security (TLS) aktivieren
  - Aktivieren Sie die Verschlüsselung des Datenverkehrs zwischen Diensten AWS App Mesh mithilfe von ACM-Zertifikaten oder vom Kunden bereitgestellten Zertifikaten
  - Schrittweise Anleitung für TLS ACM
  - Schrittweise Anleitung für TLS-Dateien
  - Envoy
- Verwenden von Nitro-Instances:

  Standardmäßig wird der Datenverkehr zwischen den folgenden Nitro-Instance-Typen automatisch verschlüsselt: C5n, G4, I3en, M5dn, M5n, P3dn, R5dn, und R5n. Der Datenverkehr wird nicht verschlüsselt, wenn er über ein Transit-Gateway, einen Load Balancer oder einen ähnlichen Vermittler geleitet wird.

  - Verschlüsselung während der Übertragung
  - Was ist die neue Ankündigung von 2019
  - Dieser Vortrag von Re:inForce 2019
- Server Name Indication (SNI) mit einem Application Load Balancer verwenden:

  Der Application Load Balancer (ALB) und der Network Load Balancer (NLB) unterstützen Server Name Indication (SNI). Mithilfe von SNI können Sie mehrere sichere Anwendungen hinter einem einzigen Listener platzieren. Hierfür hat jeder sein eigenes TLS-Zertifikat. Wir empfehlen, dass Sie Zertifikate für den Load Balancer mithilfe von AWS Certificate Manager (ACM) bereitstellen und sie dann zur Zertifikatsliste des Listeners hinzufügen. Der AWS Load Balancer verwendet einen intelligenten Algorithmus zur Zertifikatsauswahl mit SNI. Wenn der von einem Client bereitgestellte Hostname nur mit einem Zertifikat in der Zertifikatliste übereinstimmt, wählt der Load

Balancer dieses Zertifikat aus. Wenn ein von einem Client bereitgestellter Hostname mit mehreren Zertifikaten in der Liste übereinstimmt, wählt der Load Balancer ein Zertifikat aus, das der Client unterstützen kann. Beispiele hierfür sind ein selbstsigniertes Zertifikat oder ein über das ACM generiertes Zertifikat.

- SNI mit Application Load Balancer
- SNI mit Network Load Balancer

- nd-to-end E-Verschlüsselung mit TLS-Zertifikaten:

  Dies beinhaltet die Bereitstellung eines TLS-Zertifikats für die Aufgabe. Dabei kann es sich entweder um ein selbstsigniertes Zertifikat oder um ein Zertifikat einer vertrauenswürdigen Zertifizierungsstelle handeln. Sie können das Zertifikat erhalten, indem Sie auf ein Secret für das Zertifikat verweisen. Andernfalls können Sie einen Container ausführen, der eine Certificate Signing Request (CSR) an ACM ausgibt und den resultierenden geheimen Schlüssel dann auf einem gemeinsam genutzten Volume mountet.

  - Aufrechterhaltung der Sicherheit der Transportebene bis zu Ihren Containern unter Verwendung des Network Load Balancers mit Amazon ECS, Teil 1
  - Aufrechterhaltung der Transport Layer Security (TLS) bis hin zu Ihrem Container Teil 2: Verwenden AWS Private Certificate Authority

Aufgabenvernetzung

Die folgenden Empfehlungen beziehen sich auf die Funktionsweise von Amazon ECS. Amazon ECS verwendet kein Overlay-Netzwerk. Stattdessen sind die Aufgaben so konfiguriert, dass sie in verschiedenen Netzwerkmodi arbeiten. Aufgaben, die für die Verwendung des `bridge`-Modus konfiguriert sind, erhalten beispielsweise eine nicht weiterleitbare IP-Adresse von einem Docker-Netzwerk, das auf jedem Host läuft. Aufgaben, die für die Verwendung des `awsvpc`-Netzwerkmodus konfiguriert sind, beziehen eine IP-Adresse aus dem Subnetz des Hosts. Aufgaben, die für `host`-Netzwerke konfiguriert sind, verwenden die Netzwerkschnittstelle des Hosts. `awsvpc` ist der bevorzugte Netzwerkmodus. Dies liegt daran, dass dies der einzige Modus ist, mit dem Sie Sicherheitsgruppen an Aufgaben zuweisen können. Es ist auch der einzige Modus, der für AWS Fargate Aufgaben auf Amazon ECS verfügbar ist.

Sicherheitsgruppen für Aufgaben

Es wird empfohlen, Ihre Aufgaben zur Verwendung des `awsvpc`-Netzwerkmodus zu konfigurieren. Nachdem Sie Ihre Aufgabe für die Verwendung dieses Modus konfiguriert haben, stellt der Amazon-ECS-Agent automatisch eine Elastic-Network-Schnittstelle (ENI) bereit und fügt sie der

Aufgabe hinzu. Wenn die ENI bereitgestellt wird, wird die Aufgabe in einer AWS Sicherheitsgruppe registriert. Die Sicherheitsgruppe dient als virtuelle Firewall, mit der Sie den ein- und ausgehenden Datenverkehr kontrollieren können.

AWS PrivateLink und Amazon ECS

AWS PrivateLink ist eine Netzwerktechnologie, mit der Sie private Endpunkte für verschiedene AWS Dienste, einschließlich Amazon ECS, einrichten können. Die Endpunkte sind in Sandbox-Umgebungen erforderlich, in denen kein Internet-Gateway (IGW) an die Amazon VPC angeschlossen ist und keine alternativen Routen zum Internet bestehen. Durch die Verwendung AWS PrivateLink wird sichergestellt, dass Aufrufe an den Amazon ECS-Service innerhalb der Amazon VPC bleiben und nicht das Internet durchqueren. Anweisungen zum Erstellen von AWS PrivateLink Endpunkten für Amazon ECS und andere verwandte Services finden Sie unter Amazon ECS-Schnittstelle Amazon VPC-Endpoints.

> **Important**
>
> AWS Fargate Aufgaben erfordern keinen AWS PrivateLink Endpunkt für Amazon ECS.

Sowohl Amazon ECR als auch Amazon ECS unterstützen Endpunktrichtlinien. Mit diesen Richtlinien können Sie den Zugriff auf die APIs eines Services verfeinern. Sie könnten beispielsweise eine Endpunktrichtlinie für Amazon ECR erstellen, nach der nur Images an Registrys bestimmter AWS -Konten gesendet werden können. Eine Richtlinie wie diese könnte verwendet werden, um zu verhindern, dass Daten durch Container-Images exfiltriert werden, während Benutzer weiterhin die Möglichkeit haben, an autorisierte Amazon-ECR-Registrys zu senden. Weitere Informationen finden Sie unter VPC-Endpunkt-Richtlinien verwenden.

Die folgende Richtlinie ermöglicht es allen AWS Principals in Ihrem Konto, alle Aktionen nur für Ihre Amazon ECR-Repositorys durchzuführen:

```
{
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    },
```

```
  ]
}
```

Sie können dies weiter verbessern, indem Sie eine Bedingung festlegen, die die neue `PrincipalOrgID`-Eigenschaft verwendet. Dadurch wird verhindert, dass ein IAM-Principal, der nicht Teil Ihres IAM-Principals ist, das Push und Pull von Bildern durchführt. AWS Organizations Weitere Informationen finden Sie unter aws: PrincipalOrg ID.

Wir haben empfohlen, dieselbe Richtlinie sowohl auf die `com.amazonaws.region.ecr.dkr`- als auch die `com.amazonaws.region.ecr.api`-Endpunkte anzuwenden.

Einstellungen für den Container-Agenten

Die Konfigurationsdatei des Amazon ECS Container-Agenten enthält mehrere Umgebungsvariablen, die sich auf die Netzwerksicherheit beziehen. `ECS_AWSVPC_BLOCK_IMDS` und `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` werden verwendet, um den Zugriff einer Aufgabe zu Amazon EC2-Metadaten zu blockieren. `HTTP_PROXY` wird verwendet, um den Agenten so zu konfigurieren, dass er über einen HTTP-Proxy eine Verbindung zum Internet herstellt. Anweisungen zur Konfiguration des Agenten und der Docker-Laufzeit für die Weiterleitung über einen Proxy finden Sie unter HTTP-Proxykonfiguration.

> **Important**
>
> Diese Einstellungen sind nicht verfügbar, wenn Sie sie AWS Fargate verwenden.

Empfehlungen zur Netzwerksicherheit

Wir empfehlen Ihnen, bei der Einrichtung Ihrer Amazon VPC, Load Balancer und Ihres Netzwerks wie folgt vorzugehen.

Verwenden Sie gegebenenfalls Netzwerkverschlüsselung mit Amazon ECS

Sie sollten gegebenenfalls Netzwerkverschlüsselung verwenden. Bestimmte Compliance-Programme, wie PCI DSS, verlangen, dass Sie Daten während der Übertragung verschlüsseln, wenn die Daten Karteninhaberdaten enthalten. Wenn Ihr Workload ähnliche Anforderungen hat, konfigurieren Sie die Netzwerkverschlüsselung.

Moderne Browser warnen Benutzer, wenn sie eine Verbindung zu unsicheren Websites herstellen. Wenn Ihr Service von einem öffentlich zugänglichen Load Balancer unterstützt wird, verwenden Sie

TLS/SSL, um den Datenverkehr vom Browser des Clients zum Load Balancer zu verschlüsseln, und verschlüsseln Sie ihn gegebenenfalls erneut zum Backend.

Verwenden Sie den **`awsvpc`** Netzwerkmodus und Sicherheitsgruppen, um den Verkehr zwischen Aufgaben und anderen Ressourcen in Amazon ECS zu steuern

Sie sollten den `awsvpc`-Netzwerkmodus und Sicherheitsgruppen verwenden, wenn Sie den Verkehr zwischen Aufgaben sowie solchen zwischen Aufgaben und anderen Netzwerkressourcen kontrollieren müssen. Wenn sich Ihr Service hinter einem ALB befindet, verwenden Sie Sicherheitsgruppen, um nur eingehenden Datenverkehr von anderen Netzwerkressourcen zuzulassen, die die gleiche Sicherheitsgruppe wie Ihr ALB verwenden. Wenn sich Ihre Anwendung hinter einem NLB befindet, konfigurieren Sie die Sicherheitsgruppe der Aufgabe so, dass nur eingehender Datenverkehr aus dem Amazon-VPC-CIDR-Bereich und den statischen IP-Adressen, die dem NLB zugewiesen sind, zugelassen wird.

Sicherheitsgruppen sollten auch verwendet werden, um den Datenverkehr zwischen Aufgaben und anderen Ressourcen innerhalb der Amazon VPC wie beispielsweise Amazon-RDS-Datenbanken zu kontrollieren.

Erstellen Sie Amazon ECS-Cluster in separaten Amazon VPCs, wenn der Netzwerkverkehr strikt isoliert werden muss

Sie sollten Cluster in separaten Amazon VPCs erstellen, wenn der Netzwerkverkehr strikt isoliert werden muss. Vermeiden Sie es, Workloads mit strengen Sicherheitsanforderungen auf Clustern mit Workloads auszuführen, die diese Anforderungen nicht erfüllen müssen. Wenn eine strikte Netzwerkisolierung erforderlich ist, erstellen Sie Cluster in separaten Amazon VPCs und stellen Sie Services mithilfe von Amazon-VPC-Endpunkten selektiv anderen Amazon VPCs zur Verfügung. Weitere Informationen finden Sie unter Amazon-VPC-Endpunkte.

Konfiguration von AWS PrivateLink Endpunkten, sofern für Amazon ECS eine Garantie besteht

Sie sollten Endpunkte und AWS PrivateLink Endpunkte konfigurieren, sofern dies gerechtfertigt ist. Wenn Ihre Sicherheitsrichtlinie Sie daran hindert, ein Internet Gateway (IGW) an Ihre Amazon VPCs anzuhängen, konfigurieren Sie AWS PrivateLink Endpunkte für Amazon ECS und andere Dienste wie Amazon ECR, und Amazon. AWS Secrets Manager CloudWatch

Verwenden Sie Amazon VPC Flow Logs, um den Verkehr zu und von lang andauernden Aufgaben in Amazon ECS zu analysieren

Sie sollten Amazon-VPC-Flow-Protokolle verwenden, um den Verkehr zu und von lang andauernden Aufgaben zu analysieren. Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, erhalten ihre

eigene ENI. Auf diese Weise können Sie mithilfe von Amazon-VPC-Flow-Protokollen den Verkehr überwachen, der zu und von einzelnen Aufgaben geht. Ein kürzlich veröffentlichtes Update von Amazon-VPC-Flow-Protokolle (v3) bereichert die Protokolle mit Verkehrsmetadaten wie der VPC-ID, der Subnetz-ID und der Instance-ID. Diese Metadaten können verwendet werden, um eine Untersuchung einzugrenzen. Weitere Informationen finden Sie unter Amazon-VPC-Flow-Protokolle.

> **Note**
>
> Aufgrund des temporären Charakters von Containern sind Flow-Protokolle möglicherweise nicht immer eine effektive Methode, um Verkehrsmuster zwischen verschiedenen Containern oder Containern und anderen Netzwerkressourcen zu analysieren.

Netzwerkoptionen für Amazon ECS-Aufgaben für den EC2-Starttyp

Das Netzwerkverhalten von Amazon-ECS-Aufgaben, die auf Amazon-EC2-Instances gehostet werden, hängt vom Netzwerkmodus ab, der in der Aufgabendefinition definiert ist. Wir empfehlen die Verwendung des `awsvpc`-Netzwerkmodus, es sei denn, Sie müssen einen anderen Netzwerkmodus verwenden.

Im Folgenden sind die verfügbaren Netzwerkmodi aufgeführt.

Netzwerkmodus	Linux-Container auf EC2	Windows-Containers auf EC2	Beschreibung
`awsvpc`	Ja	Ja	Der Aufgabe wird eine eigene Elastic-Network-Schnittstelle (ENI) und eine primäre private IPv4-Adresse zugewiesen. Dadurch erhalten die Aufgabe dieselben Netzwerkeigenschaften wie Amazon-EC2-Instances.
`bridge`	Ja	Nein	Die Aufgabe verwendet das integrierte virtuelle Netzwerk von Docker unter Linux, das innerhalb jeder Amazon EC2-Instance ausgeführt wird, die die Aufgabe hostet. Das integrierte virtuelle Netzwerk unter Linux verwendet den `bridge` Docker-Netzwerktre

Netzwerkmodus	Linux-Container auf EC2	Windows-Containers auf EC2	Beschreibung
			iber. Dies ist der Standard-Netzwerkmodus unter Linux, wenn in der Aufgabendefinition kein Netzwerkmodus angegeben ist.
`host`	Ja	Nein	Die Aufgabe verwendet das Hostnetzwerk, das das integrierte virtuelle Netzwerk von Docker umgeht, indem es Container-Ports direkt der ENI der Amazon EC2-Instance zuweist, die die Aufgabe hostet. Dynamische Portzuordnungen können in diesem Netzwerkmodus nicht verwendet werden. Ein Container in einer Aufgabendefinition, der diesen Modus verwendet, muss eine bestimmte `hostPort`-Nummer angeben. Eine Portnummer auf einem Host kann nicht für mehrere Aufgaben verwendet werden. Daher können Sie nicht mehrere Aufgaben derselben Aufgabendefinition auf einer einzelnen Amazon EC2-Instance ausführen.
`none`	Ja	Nein	Die Aufgabe verfügt über keine externe Netzwerkverbindung.
`default`	Nein	Ja	Die Aufgabe verwendet das integrierte virtuelle Netzwerk von Docker unter Windows, das innerhalb jeder Amazon EC2-Instance ausgeführt wird, die die Aufgabe hostet. Das integrierte virtuelle Netzwerk unter Windows verwendet den `nat` Docker-Netzwerktreiber. Dies ist der Standard-Netzwerkmodus unter Windows, wenn in der Aufgabendefinition kein Netzwerkmodus angegeben ist.

Weitere Informationen zum Docker-Netzwerk finden Sie in der Docker-Dokumentation unter Netzwerkübersicht.

Weitere Informationen zu Docker-Netzwerken unter Windows finden Sie unter Windows-Containernetzwerke in der Dokumentation von Microsoft zu Containern unter Windows.

Zuweisen einer Netzwerkschnittstelle für eine Amazon ECS-Aufgabe

Die Aufgabenvernetzungsfeatures, die durch den Netzwerkmodus `awsvpc` bereitgestellt werden, geben Amazon-ECS-Aufgaben dieselben Netzwerkeigenschaften wie Amazon-EC2-Instances. Die Verwendung des `awsvpc` Netzwerkmodus vereinfacht Container-Netzwerke, da Sie mehr Kontrolle darüber haben, wie Ihre Anwendungen miteinander und mit anderen Diensten innerhalb Ihrer VPCs kommunizieren. Der `awsvpc` Netzwerkmodus bietet auch mehr Sicherheit für Ihre Container, da Sie innerhalb Ihrer Aufgaben Sicherheitsgruppen und Netzwerküberwachungstools auf einer detaillierteren Ebene verwenden können. Sie können auch andere Amazon EC2 EC2-Netzwerkfunktionen wie VPC Flow Logs verwenden, um den Verkehr zu und von Ihren Aufgaben zu überwachen. Außerdem können Container, die zur selben Aufgabe gehören, über die Schnittstelle `localhost`kommunizieren.

Das Task elastic network interface (ENI) ist eine vollständig verwaltete Funktion von Amazon ECS. Amazon ECS erstellt die ENI und hängt sie an die Amazon EC2 Host-Instance mit der angegebenen Sicherheitsgruppe an. Die Aufgabe sendet und empfängt Netzwerkverkehr über die ENI auf dieselbe Weise wie Amazon-EC2-Instances mit ihren primären Netzwerkschnittstellen. Jeder Aufgaben-ENI wird standardmäßig eine private IPv4-Adresse zugewiesen. Wenn Ihre VPC für den Dual-Stack-Modus aktiviert ist und Sie ein Subnetz mit einem IPv6-CIDR-Block verwenden, erhält die Aufgaben-ENI auch eine IPv6-Adresse. Jede Aufgabe kann nur eine ENI haben.

Diese ENIs sind in der Amazon EC2 EC2-Konsole für Ihr Konto sichtbar. Ihr Konto kann die ENIs nicht trennen oder ändern. Auf diese Weise wird ein versehentliches Löschen einer ENI, die der aktuell ausgeführten Aufgabe zugeordnet wird, verhindert. Sie können die ENI-Anhangsinformationen für Aufgaben in der Amazon ECS-Konsole oder bei der DescribeTasksAPI-Operation anzeigen. Wenn die Aufgabe angehalten oder der Service herunterskaliert wird, wird die Aufgaben-ENI getrennt und gelöscht.

Wenn Sie eine höhere ENI-Dichte benötigen, verwenden Sie die `awsvpcTrunking` Kontoeinstellungen. Amazon ECS erstellt auch eine „Trunk“ -Netzwerkschnittstelle für Ihre Container-Instance und fügt sie an. Das Stamm-Netzwerk wird vollständig von Amazon ECS verwaltet. Die Stamm-ENI wird gelöscht, wenn Sie Ihre Container-Instance beenden oder die Registrierung auf

dem Amazon-ECS-Cluster aufheben. Weitere Informationen zur `awsvpcTrunking` Kontoeinstellung finden Sie unterVoraussetzungen.

Sie geben `awsvpc` dies im `networkMode` Parameter der Aufgabendefinition an. Weitere Informationen finden Sie unter Netzwerkmodus.

Wenn Sie dann eine Aufgabe ausführen oder einen Dienst erstellen, verwenden Sie den `networkConfiguration` Parameter, der ein oder mehrere Subnetze umfasst, um Ihre Aufgaben einer oder mehreren Sicherheitsgruppen zuzuordnen, um sie an eine ENI anzuhängen. Weitere Informationen finden Sie unter Netzwerkkonfiguration. Die Aufgaben werden auf vereinbaren Amazon-EC2-Instances in denselben Availability Zones wie solche Subnetze platziert und die angegebenen Sicherheitsgruppen werden der ENI zugeordnet, die für die Aufgabe bereitgestellt wird.

Überlegungen zu Linux

Beachten Sie Folgendes, wenn Sie das Linux-Betriebssystem verwenden.

- Wenn Sie eine p5.48xlarge-Instance im `awsvpc` Modus verwenden, können Sie nicht mehr als eine Aufgabe auf der Instance ausführen.
- Für Aufgaben und Dienste, die den `awsvpc` Netzwerkmodus verwenden, ist die mit dem Service verknüpfte Amazon ECS-Rolle erforderlich, um Amazon ECS die Berechtigungen zu erteilen, in Ihrem Namen Anrufe an andere AWS Dienste zu tätigen. Diese Rolle wird automatisch für Sie erstellt, wenn Sie einen Cluster erstellen oder wenn Sie einen Service in der AWS Management Console erstellen oder aktualisieren. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS. Sie können die serviceverknüpfte Rolle auch mit dem folgenden AWS CLI Befehl erstellen:

  ```
  aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
  ```

- Für Ihre Amazon EC2 Linux-Instance ist eine Version `1.15.0` oder höher des Container-Agents erforderlich, um Tasks auszuführen, die den `awsvpc`Netzwerkmodus verwenden. Wenn Sie das Amazon-ECS-optimierte AMI verwenden, benötigt Ihre Instance mindestens Version `1.15.0-4` des `ecs-init`-Pakets.
- Amazon ECS füllt den Hostnamen der Aufgabe mit einem von Amazon bereitgestellten (internen) DNS-Hostnamen auf, wenn sowohl die Option `enableDnsHostnames` als auch die Option `enableDnsSupport` auf Ihrer VPC aktiviert sind. Wenn diese Optionen nicht aktiviert sind, ist der DNS-Hostname der Aufgabe ein zufälliger Hostname. Weitere Informationen zu den DNS-Einstellungen für eine ECS finden Sie unter Verwenden von DNS in Ihrer VPC im Amazon-VPC-Benutzerhandbuch.

- Jede Amazon-ECS-Aufgabe, die den Netzwerkmodus `awsvpc` verwendet, erhält eine eigene Elastic-Network-Schnittstelle (ENI), die an die Amazon-EC2-Instance angehängt wird, die sie hostet. Es gibt ein Standardkontingent für die Anzahl der Netzwerkschnittstellen, die an eine Amazon-EC2-Linux-Instance angehängt werden können. Dabei zählt die primäre Netzwerkschnittstelle als eine Einheit. Beispiel: Standardmäßig können an eine `c5.large`-Instance nur bis zu drei ENIs angehängt werden. Die primäre Netzwerkschnittstelle für die Instance zählt dazu. Sie können zwei zusätzliche ENIs zu der Instance anhängen. Da jede Aufgabe, die den Netzwerkmodus `awsvpc` verwendet, eine ENI benötigt, können Sie nur zwei solcher Aufgaben auf diesem Instance-Typ ausführen. Weitere Informationen zu den standardmäßigen ENI-Grenzwerten für jeden Instance-Typ finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2 EC2-Benutzerhandbuch.
- Amazon ECS unterstützt das Starten von Amazon-EC2-Linux-Instances mit unterstützten Instance-Typen mit erhöhter ENI-Dichte. Wenn Sie sich bei der `awsvpcTrunking`-Kontoeinstellung anmelden und Amazon EC2 Linux-Instances mit diesen Instance-Typen in Ihrem Cluster registrieren, haben diese Instances ein höheres ENI-Kontingent. Wenn Sie diese Instances mit diesem höheren Kontingent verwenden, können Sie mehr Aufgaben auf jeder Amazon-EC2-Linux-Instance platzieren. Um die erhöhte ENI-Dichte mit dem Trunking-Feature zu nutzen, benötigen Ihre Amazon EC2-Instances mindestens Version `1.28.1` des Container-Agenten. Wenn Sie das Amazon-ECS-optimierte AMI verwenden, benötigt Ihre Instance mindestens Version `1.28.1-2` des `ecs-init`-Pakets. Weitere Informationen zum Aktivieren der `awsvpcTrunking`-Kontoeinstellung finden Sie unter Greifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu. Weitere Informationen zum ENI-Trunking finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.
- Beim Hosten von Aufgaben, die den `awsvpc`-Netzwerkmodus auf Amazon EC2 Linux-Instances verwenden, erhalten Ihre Aufgaben-ENIs keine öffentlichen IP-Adressen. Um auf das Internet zuzugreifen, müssen Aufgaben in einem privaten Subnetz gestartet werden, das für die Verwendung eines NAT-Gateways konfiguriert ist. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Der eingehende Netzwerkzugriff muss innerhalb einer VPC mithilfe der privaten IP-Adresse oder über einen Load Balancer innerhalb der VPC weitergeleitet werden. Aufgaben, die in öffentlichen Subnetzen gestartet werden, haben keinen Zugriff auf das Internet.
- Amazon ECS erkennt nur die ENIs, die an Ihre Amazon-EC2-Linux-Instances angehängt wurden. Wenn Sie ENIs manuell an Ihre Instances angefügt haben, könnte Amazon ECS versuchen, eine Aufgabe einer Instance hinzuzufügen, die nicht über genügend Netzwerkadapter verfügt. Das kann zu einem Timout bei der Aufgabe führen, sodass diese in den Status der Aufhebung der

Bereitstellung übergeht und dann gestoppt wird. Wir empfehlen Ihnen, keine ENIs manuell an Ihre Instances anzufügen.

- Amazon EC2 Linux-Instances müssen mit der `ecs.capability.task-eni`-Fähigkeit registriert werden, um für die Platzierung von Aufgaben im Netzwerkmodus `awsvpc` berücksichtigt zu werden. Instances, die Version `1.15.0-4` oder höher von `ecs-init` ausführen, werden automatisch mit diesem Attribut registriert.
- Die ENIs, die an Ihre Amazon EC2 Linux-Instances angefügt wurden und auch damit erstellt wurden, können nicht manuell getrennt oder über Ihr Konto geändert werden. Auf diese Weise wird ein versehentliches Löschen einer ENI, die einer aktuell ausgeführten Aufgabe zugeordnet wird, verhindert. Wenn Sie die ENIs für eine Aufgabe freigeben möchten, stoppen Sie die Aufgabe.
- Es gilt eine Beschränkung von 16 Subnetzen und 5 Sicherheitsgruppen, die in `awsVpcConfiguration` angegeben werden können, wenn eine Aufgabe ausgeführt oder ein Service erstellt wird, der den `awsvpc`-Netzwerkmodus verwendet. Weitere Informationen finden Sie unter AwsVpcKonfiguration in der Amazon Elastic Container Service API-Referenz.
- Wenn eine Aufgabe mit dem Netzwerkmodus `awsvpc` gestartet wird, erstellt der Amazon-ECS-Containeragent einen zusätzlichen `pause`-Container für jede Aufgabe, bevor er die Container in der Aufgabendefinition startet. Anschließend konfiguriert es den Netzwerk-Namespace des `pause` Containers, indem es die amazon-ecs-cni-plugins CNI-Plugins ausführt. Dann startet der Agent die restlichen Container in der Aufgabe, sodass sie den Netzwerk-Stack des `pause`-Containers gemeinsam verwenden. Das bedeutet, dass alle Container in der Aufgabe über IP-Adressen der Elastic-Network-Schnittstelle angesprochen werden und miteinander über die `localhost`-Schnittstelle kommunizieren können.
- Services mit Aufgaben, die den`awsvpc`-Netzwerkmodus verwenden, unterstützen nur Application Load Balancer und Network Load Balancer. Wenn Sie eine beliebige Zielgruppe für diese Services erstellen, müssen Sie zudem `ip` als Zieltyp auswählen. Verwenden Sie nicht `instance`. Dies liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, einer ENI zugeordnet sind, nicht einer Amazon EC2 Linux-Instance. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.
- Wenn Ihre VPC aktualisiert wird, um die eingestellten DHCP-Optionen zu ändern, können Sie diese Änderungen nicht für vorhandene Aufgaben übernehmen. Starten Sie neue Aufgaben, für die diese Änderungen übernommen wurden. Überprüfen Sie, ob sie ordnungsgemäß funktionieren, und beenden Sie dann die vorhandenen Aufgaben, um diese Netzwerkkonfigurationen sicher zu ändern.

Überlegungen zu Windows

Beachten Sie Folgendes, wenn Sie das Windows-Betriebssystem verwenden:

- Container-Instances, die das für Amazon ECS optimierte Windows Server 2016-AMI verwenden, können keine Aufgaben hosten, die den Netzwerkmodus `awsvpc` verwenden. Wenn Sie über einen Cluster verfügen, der Amazon-ECS-optimierte Windows Server 2016-AMIs und Windows-AMIs enthält, die den `awsvpc`-Netzwerkmodus unterstützen, werden Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden werden nicht auf den Windows 2016 Server-Instances gestartet. Vielmehr werden sie auf Instances gestartet, die den `awsvpc`-Netzwerkmodus unterstützen.
- Ihre Amazon EC2 Windows-Instance benötigt Version `1.57.1` oder höher des Container-Agenten, um CloudWatch Metriken für Windows-Container zu verwenden, die den `awsvpc` Netzwerkmodus verwenden.
- Für Aufgaben und Dienste, die den `awsvpc` Netzwerkmodus verwenden, ist die mit dem Service verknüpfte Amazon ECS-Rolle erforderlich, um Amazon ECS die Berechtigungen zu erteilen, in Ihrem Namen Anrufe an andere AWS Dienste zu tätigen. Diese Rolle wird automatisch für Sie erstellt, wenn Sie einen Cluster erstellen oder wenn Sie einen Service in AWS Management Console erstellen oder aktualisieren. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS. Sie können die serviceverknüpfte Rolle auch mit dem folgenden AWS CLI Befehl erstellen.

  ```
  aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
  ```

- Für Ihre Amazon EC2 Windows-Instance ist eine Version `1.54.0` des Container-Agents oder höher erforderlich, um Tasks auszuführen, die den `awsvpc`-Netzwerkmodus verwenden. Wenn Sie auf der Instance ein Bootstrap ausführen, müssen Sie die Optionen konfigurieren, die für den `awsvpc`-Netzwerkmodus erforderlich sind. Weitere Informationen finden Sie unter the section called “Bootstrapping von Container-Instances”.
- Amazon ECS füllt den Hostnamen der Aufgabe mit einem von Amazon bereitgestellten (internen) DNS-Hostnamen auf, wenn sowohl die Option `enableDnsHostnames` als auch die Option `enableDnsSupport` auf Ihrer VPC aktiviert sind. Wenn diese Optionen nicht aktiviert sind, ist der DNS-Hostname der Aufgabe ein zufälliger Hostname. Weitere Informationen zu den DNS-Einstellungen für eine ECS finden Sie unter Verwenden von DNS in Ihrer VPC im Amazon-VPC-Benutzerhandbuch.
- Jede Amazon-ECS-Aufgabe, die den Netzwerkmodus `awsvpc` verwendet, erhält eine eigene Elastic-Network-Schnittstelle (ENI), die an die Amazon-EC2-Windows-Instance angehängt wird, die sie hostet. Es gibt ein Standardkontingent für die Anzahl der Netzwerkschnittstellen, die

an eine Amazon-EC2-Windows-Instance angehängt werden können. Dabei zählt die primäre Netzwerkschnittstelle als eine Einheit. Beispiel: Standardmäßig können an eine `c5.large`-Instance nur bis zu drei ENIs angehängt werden. Die primäre Netzwerkschnittstelle für die Instance zählt dazu. Sie können zwei zusätzliche ENIs zu der Instance anhängen. Da jede Aufgabe, die den Netzwerkmodus `awsvpc` verwendet, eine ENI benötigt, können Sie nur zwei solcher Aufgaben auf diesem Instance-Typ ausführen. Weitere Informationen zu den standardmäßigen ENI-Grenzwerten für jeden Instance-Typ finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2 EC2-Benutzerhandbuch.

- Beim Hosten von Aufgaben, die den `awsvpc`-Netzwerkmodus auf Amazon EC2 Windows-Instances verwenden, erhalten Ihre Aufgaben-ENIs keine öffentlichen IP-Adressen. Um auf das Internet zuzugreifen, starten Sie Aufgaben in einem privaten Subnetz, das für die Verwendung eines NAT-Gateways konfiguriert ist. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch. Der eingehende Netzwerkzugriff muss innerhalb der VPC mithilfe der privaten IP-Adresse oder über einen Load Balancer innerhalb der VPC weitergeleitet werden. Aufgaben, die in öffentlichen Subnetzen gestartet werden, haben keinen Zugriff auf das Internet.
- Amazon ECS erkennt nur die ENIs, die an Ihre Amazon EC2 Windows-Instances angehängt wurden. Wenn Sie ENIs manuell an Ihre Instances angefügt haben, könnte Amazon ECS versuchen, eine Aufgabe einer Instance hinzuzufügen, die nicht über genügend Netzwerkadapter verfügt. Das kann zu einem Timout bei der Aufgabe führen, sodass diese in den Status der Aufhebung der Bereitstellung übergeht und dann gestoppt wird. Wir empfehlen Ihnen, keine ENIs manuell an Ihre Instances anzufügen.
- Amazon EC2-Windows-Instances müssen mit der `ecs.capability.task-eni`-Fähigkeit registriert werden, um für die Platzierung von Aufgaben im Netzwerkmodus `awsvpc` berücksichtigt zu werden.
- Sie können ENIs, die an Ihre Amazon EC2-Windows-Instances angefügt wurden und auch damit erstellt wurden, nicht manuell trennen oder ändern. Auf diese Weise wird ein versehentliches Löschen einer ENI, die der aktuell ausgeführten Aufgabe zugeordnet wird, verhindert. Wenn Sie die ENIs für eine Aufgabe freigeben möchten, stoppen Sie die Aufgabe.
- Sie können nur bis zu 16 Subnetze und 5 Sicherheitsgruppen in `awsVpcConfiguration` angeben, wenn Sie Aufgaben ausführen oder Services erstellen, die den `awsvpc`-Netzwerkmodus verwenden. Weitere Informationen finden Sie unter AwsVpcKonfiguration in der Amazon Elastic Container Service API-Referenz.
- Wenn eine Aufgabe mit dem Netzwerkmodus `awsvpc` gestartet wird, erstellt der Amazon-ECS-Containeragent einen zusätzlichen `pause`-Container für jede Aufgabe, bevor er die Container in

der Aufgabendefinition startet. Anschließend konfiguriert es den Netzwerk-Namespace des `pause` Containers, indem es die amazon-ecs-cni-plugins CNI-Plugins ausführt. Dann startet der Agent die restlichen Container in der Aufgabe, sodass sie den Netzwerk-Stack des `pause`-Containers gemeinsam verwenden. Das bedeutet, dass alle Container in der Aufgabe über IP-Adressen der Elastic-Network-Schnittstelle angesprochen werden und miteinander über die `localhost`-Schnittstelle kommunizieren können.

- Services mit Aufgaben, die den`awsvpc`-Netzwerkmodus verwenden, unterstützen nur Application Load Balancer und Network Load Balancer. Wenn Sie eine beliebige Zielgruppe für diese Services erstellen, müssen Sie `ip` als Zieltyp auswählen, und nicht `instance`. Dies liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, einer ENI zugeordnet sind, nicht einer Amazon EC2 Windows-Instance. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.
- Wenn Ihre VPC aktualisiert wird, um die eingestellten DHCP-Optionen zu ändern, können Sie diese Änderungen nicht für vorhandene Aufgaben übernehmen. Starten Sie neue Aufgaben, für die diese Änderungen übernommen wurden. Überprüfen Sie, ob sie ordnungsgemäß funktionieren, und beenden Sie dann die vorhandenen Aufgaben, um diese Netzwerkkonfigurationen sicher zu ändern.
- Die folgenden Punkte werden nicht unterstützt, wenn Sie den Netzwerkmodus `awsvpc` in einer EC2-Windows-Konfiguration verwenden:
  - Dual-Stack-Konfiguration
  - IPv6
  - ENI Trunking

Verwenden einer VPC im Dual-Stack-Modus

Wenn Sie eine VPC im Dual-Stack-Modus verwenden, können Ihre Aufgaben über IPv4 oder IPv6 oder beides kommunizieren. IPv4- und IPv6-Adressen sind unabhängig voneinander. Daher müssen Sie Routing und Sicherheit in Ihrer VPC separat für IPv4 und IPv6 konfigurieren. Weitere Informationen zum Konfigurieren der VPC für den Dual-Stack-Modus finden Sie unter Migrieren zu IPv6 im Amazon-VPC-Benutzerhandbuch.

Wenn Sie Ihre Virtual Private Cloud (VPC) mit einem Internet-Gateway oder einem reinen Outbound-Internet-Gateway konfiguriert haben, können Sie Ihre VPC im Dual-Stack-Modus verwenden. So können Aufgaben, denen eine IPv6-Adresse zugewiesen ist, über ein Internet-Gateway oder ein reines Internet-Gateway auf das Internet zugreifen. NAT-Gateways sind optional. Weitere

Informationen finden Sie unter Internet-Gateways und Internet-Gateways nur für Egress im Amazon VPC-Benutzerhandbuch.

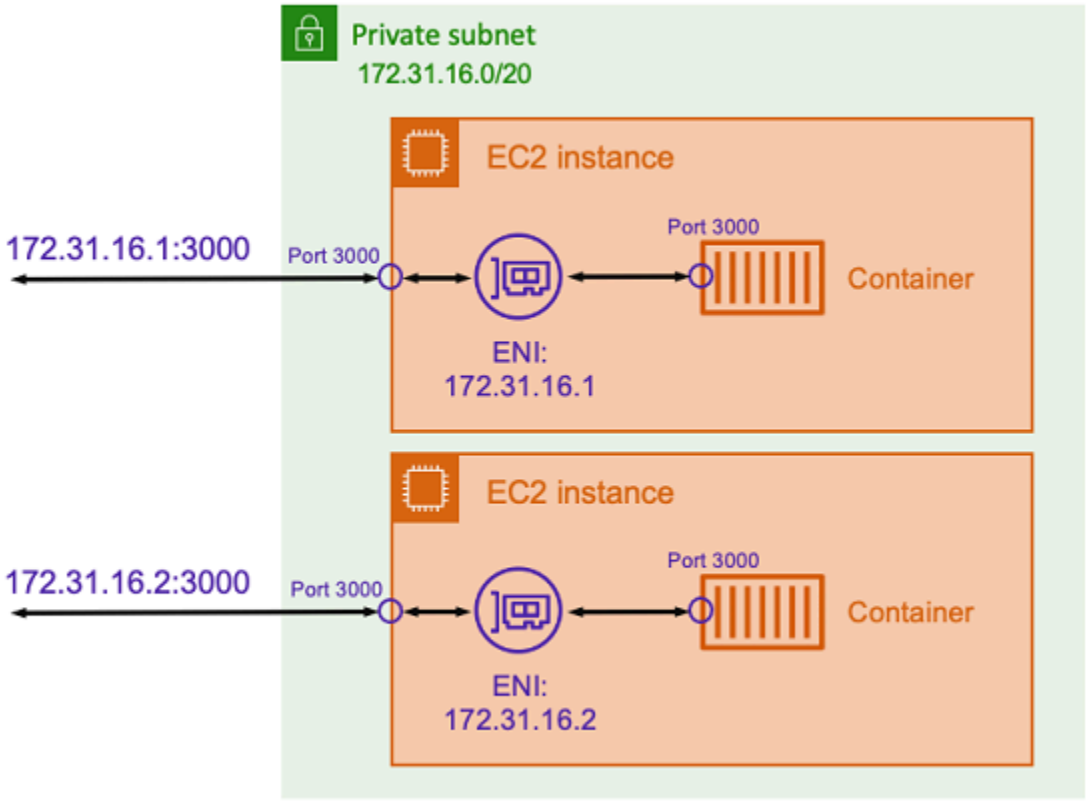
Amazon-ECS-Aufgaben werden eine IPv6-Adresse zugewiesen, wenn die folgenden Bedingungen erfüllt sind:

- Die Amazon-EC2-Linux-Instance, die die Aufgabe hostet, verwendet Version `1.45.0` oder höher des Container-Agenten. Informationen zum Überprüfen der Agent-Version, die Ihre Instance verwendet, und bei Bedarf aktualisieren, finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.
- Die Kontoeinstellung `dualStackIPv6` ist aktiviert. Weitere Informationen finden Sie unter Greifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu.
- Ihre Aufgabe verwendet den Netzwerkmodus `awsvpc`.
- Ihre VPC und Ihr Subnetz sind für IPv6 konfiguriert. Die Konfiguration enthält die Netzwerkschnittstellen, die im angegebenen Subnetz erstellt werden. Weitere Informationen zum Konfigurieren der VPC für den Dual-Stack-Modus finden Sie unter Migrieren zu IPv6 und Ändern des IPv6-Adressierungsattributs Ihres Subnetzes im Amazon-VPC-Benutzerhandbuch.

Ordnen Sie Amazon ECS-Container-Ports der EC2-Instance-Netzwerkschnittstelle zu

Der `host`-Netzwerkmodus wird nur für Amazon-ECS-Aufgaben unterstützt, die auf Amazon-EC2-Instances gehostet werden. Bei der Verwendung von Amazon ECS auf Fargate wird dies nicht unterstützt.

Der `host`-Netzwerkmodus ist der einfachste Netzwerkmodus, der in Amazon ECS unterstützt wird. Im Host-Modus ist das Netzwerk des Containers direkt an den zugrunde liegenden Host gebunden, auf dem der Container ausgeführt wird.



Gehen Sie davon aus, dass Sie einen Node.js-Container mit einer Express-Anwendung ausführen, die Listener auf einem `3000`-Port ist, der dem im vorherigen Diagramm dargestellten ähnelt. Wenn der `host`-Netzwerkmodus verwendet wird, empfängt der Container Datenverkehr auf Port 3000 unter Verwendung der IP-Adresse der zugrunde liegenden Host-Amazon-EC2-Instance. Wir raten davon ab, diesen Modus zu verwenden.

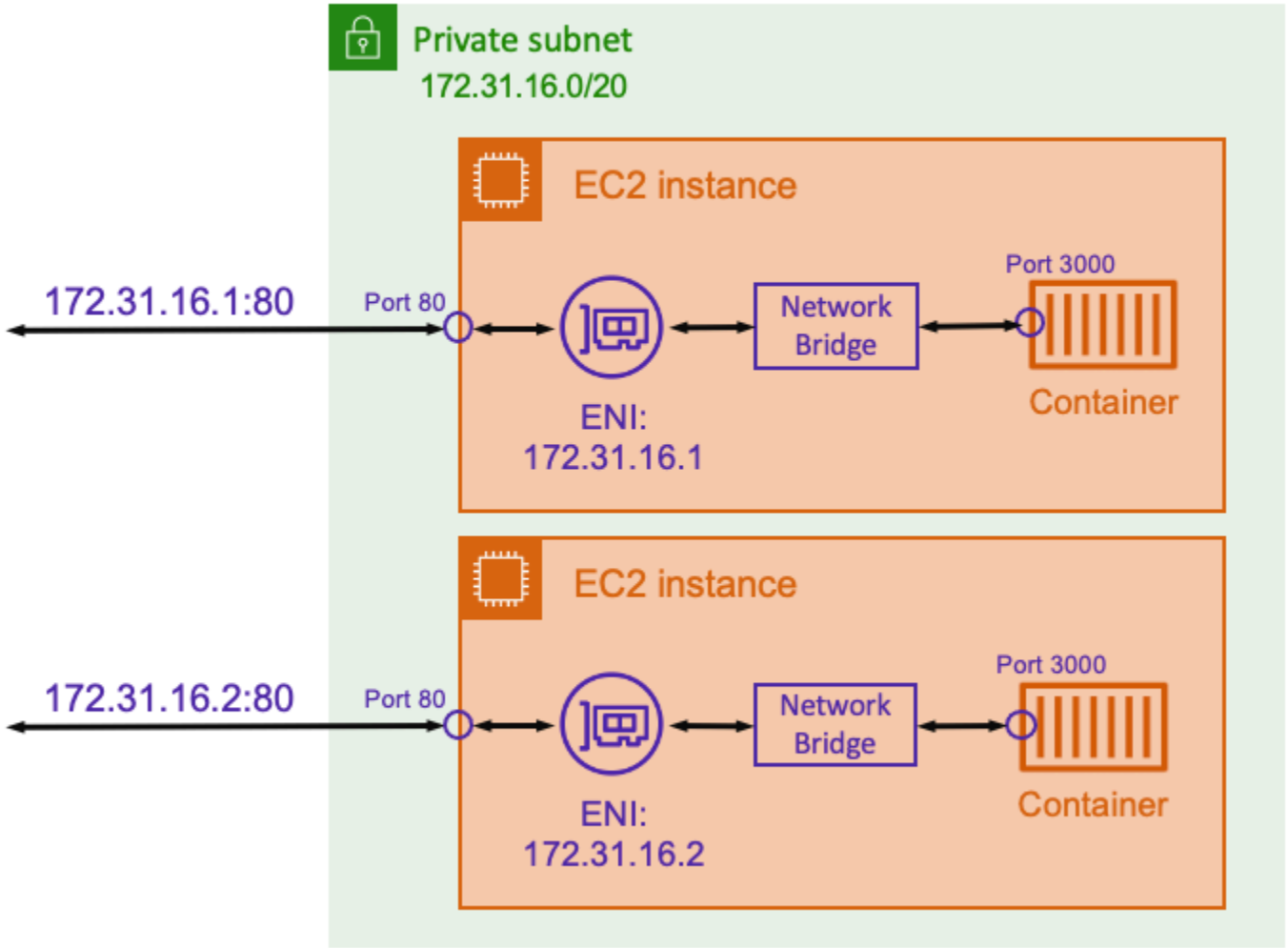
Die Verwendung dieses Netzwerkmodus hat erhebliche Nachteile. Sie können auf jedem Host nur eine einzige Instanziierung eines Tasks ausführen. Dies liegt daran, dass nur die erste Aufgabe an den erforderlichen Port auf der Amazon-EC2-Instance gebunden werden kann. Es gibt auch keine Möglichkeit, einen Container-Port neu zuzuordnen, wenn er den `host`-Netzwerkmodus verwendet. Wenn eine Anwendung beispielsweise Listener auf einer bestimmten Portnummer sein muss, können Sie die Portnummer nicht direkt neu zuordnen. Stattdessen müssen Sie alle Portkonflikte lösen, indem Sie die Anwendungskonfiguration ändern.

Die Verwendung des `host`-Netzwerkmodus hat auch Auswirkungen auf die Sicherheit. In diesem Modus können Container die Identität des Hosts annehmen und Container können sich mit privaten Loopback-Netzwerkservices auf dem Host verbinden.

Verwenden Sie das virtuelle Netzwerk von Docker für Amazon ECS-Linux-Aufgaben

Der `bridge`-Netzwerkmodus wird nur für Amazon-ECS-Aufgaben unterstützt, die auf Amazon-EC2-Instances gehostet werden.

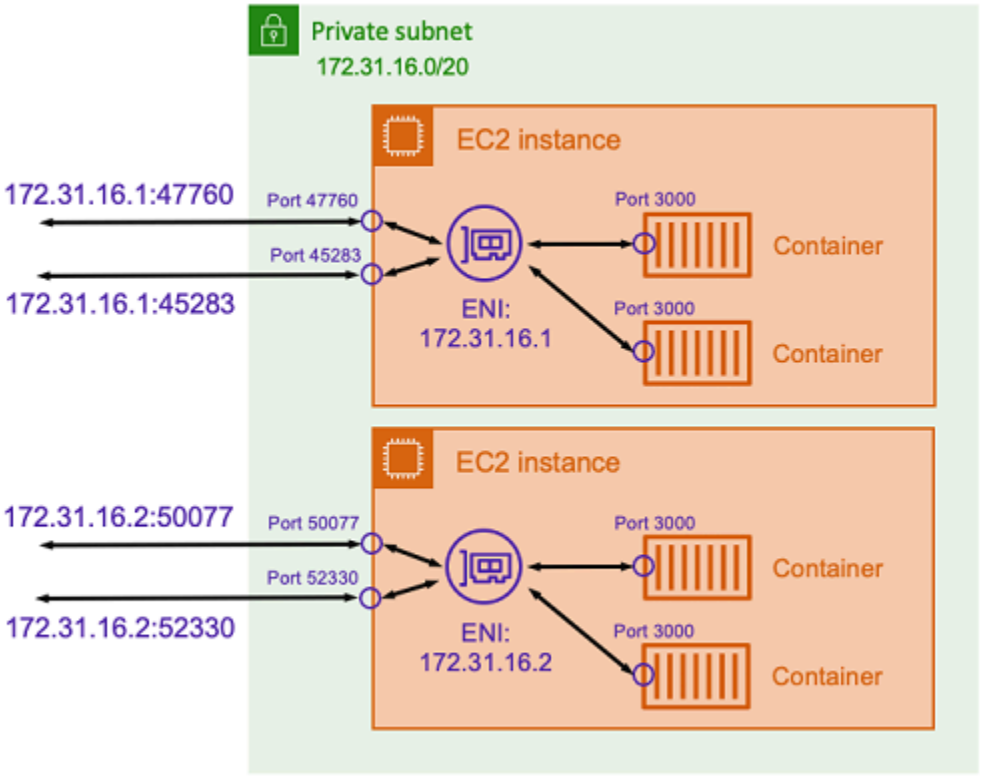
Im `bridge`-Modus verwenden Sie eine virtuelle Netzwerkbrücke, um eine Ebene zwischen dem Host und dem Netzwerk des Containers zu erstellen. Auf diese Weise können Sie Portzuordnungen erstellen, die einen Host-Port einem Container-Port neu zuordnen. Die Zuordnungen können statisch oder dynamisch sein.



Mit einer statischen Port-Zuordnung können Sie explizit definieren, welchen Host-Port Sie einem Container-Port zuordnen möchten. Im obigen Beispiel wird der Port `80` auf dem Host dem Port `3000` auf dem Container zugeordnet. Um mit der containerisierten Anwendung zu kommunizieren, senden Sie den Datenverkehr an Port `80` an die IP-Adresse der Amazon-EC2-Instance. Aus der Sicht der containerisierten Anwendung sieht sie den eingehenden Datenverkehr an Port `3000`.

Wenn Sie nur den Datenverkehr-Port ändern möchten, eignen sich statische Port-Zuordnungen. Dies hat jedoch immer noch den gleichen Nachteil wie die Verwendung des `host`-Netzwerkmodus. Sie können auf jedem Host nur eine einzige Instanziierung einer Aufgabe ausführen. Dies liegt daran, dass bei einer statischen Port-Zuordnung nur ein einziger Container Port 80 zugeordnet werden kann.

Um dieses Problem zu lösen, sollten Sie erwägen, den `bridge`-Netzwerkmodus mit einer dynamischen Portzuweisung zu verwenden, wie in der folgenden Abbildung dargestellt.



Wenn Sie in der Port-Zuordnung keinen Host-Port angeben, können Sie Docker veranlassen, einen zufälligen, ungenutzten Port aus dem flüchtigen Portbereich auszuwählen und ihn als öffentlichen Host-Port für den Container zuzuweisen. Beispielsweise könnte der Anwendung Node.js, die den Port `3000` auf dem Container überwacht, ein zufälliger Port mit hoher Nummer zugewiesen werden, z. B. `47760` auf dem Amazon-EC2-Host. Dies bedeutet, dass Sie mehrere Kopien dieses Containers auf dem Host ausführen können. Darüber hinaus kann jedem Container ein eigener Port auf dem Host zugewiesen werden. Jede Kopie des Containers empfängt Datenverkehr über Port `3000`. Clients, die Datenverkehr an diese Container senden, verwenden jedoch die nach dem Zufallsprinzip zugewiesenen Host-Ports.

Amazon ECS hilft Ihnen dabei, den Überblick über die zufällig zugewiesenen Ports für jede Aufgabe zu behalten. Zu diesem Zweck werden die Zielgruppen des Load Balancers und die AWS Cloud Map Diensterkennung automatisch aktualisiert, sodass die Liste der IP-Adressen und Ports für Aufgaben angezeigt wird. Dies erleichtert die Nutzung von Services, die im `bridge`-Modus mit dynamischen Ports betrieben werden.

Ein Nachteil der Verwendung des `bridge`-Netzwerkmodus besteht jedoch darin, dass es schwierig ist, die Kommunikation zwischen Services zu sperren. Da Services jedem beliebigen, ungenutzten Port zugewiesen werden können, ist es notwendig, breite Portbereiche zwischen Hosts zu öffnen. Es ist jedoch nicht einfach, spezifische Regeln zu erstellen, sodass ein bestimmter Service nur mit einem bestimmten anderen Service kommunizieren kann. Die Services haben keine spezifischen Ports, die für Netzwerkregeln für Sicherheitsgruppen verwendet werden können.

Netzwerkoptionen für Amazon ECS-Aufgaben für den Starttyp Fargate

Standardmäßig wird jede Amazon-ECS-Aufgabe auf Fargate eine Elastic-Network-Schnittstelle (ENI) mit einer primären privaten IP-Adresse bereitgestellt. Wenn Sie ein öffentliches Subnetz verwenden, können Sie der ENI der Aufgabe optional eine öffentliche IP-Adresse zuweisen. Wenn Ihre VPC für den Dual-Stack-Modus konfiguriert ist und Sie ein Subnetz mit einem IPv6-CIDR-Block verwenden, erhält das ENI Ihrer Aufgabe auch eine IPv6-Adresse. Einer Aufgabe kann immer nur eine ENI auf einmal zugeordnet sein. Container, die zur selben Aufgabe gehören, können auch über die `localhost`-Schnittstelle kommunizieren. Weitere Informationen über VPCs und Subnetze finden Sie unter VPC und Subnetze im Amazon VPC-Benutzerhandbuch.

Damit eine Aufgabe auf Fargate ein Container-Image abrufen kann, muss die Aufgabe eine Route zum Internet haben. Nachfolgend finden Sie Informationen darüber, wie Sie sicherstellen können, dass Ihre Aufgabe über einen Pfad zum Internet verfügt.

- Wenn Sie ein öffentliches Subnetz verwenden, können Sie der Aufgaben-ENI eine öffentliche IP-Adresse zuweisen.
- Bei Verwendung eines privaten Subnetzes kann ein NAT-Gateway an das Subnetz angeschlossen sein.
- Wenn Sie Container-Images verwenden, die in Amazon ECR gehostet werden, können Sie Amazon ECR so konfigurieren, dass ein Interface-VPC-Endpunkt verwendet wird. Der Image-Pull erfolgt über die private IPv4-Adresse der Aufgabe. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.

Da jede Aufgabe ihre eigene ENI erhält, können Sie Netzwerkfeatures wie VPC Flow Logs nutzen, sodass Sie den Verkehr zu und von Ihren Aufgaben überwachen können. Weitere Informationen finden Sie unter VPC-Flow-Protokolle im Amazon-VPC-Benutzerhandbuch.

Sie können auch die Vorteile nutzen. AWS PrivateLink Sie können einen VPC-Schnittstellenendpunkt so konfigurieren, dass Sie über private IP-Adressen auf Amazon ECS-APIs zugreifen können. AWS PrivateLink schränkt den gesamten Netzwerkverkehr zwischen Ihrer VPC und Amazon ECS auf das Amazon-Netzwerk ein. Sie benötigen kein Internet-Gateway, kein NAT-Gerät und kein Virtual Private Gateway. Weitere Informationen finden Sie unter AWS PrivateLink im Leitfaden zu bewährten Methoden für Amazon ECS.

Beispiele für die Verwendung der `NetworkConfiguration` Ressource mit finden Sie AWS CloudFormation unter. the section called “Erstellen von Amazon ECS-Ressourcen über separate Stapel”

Die erstellten ENIs werden vollständig von AWS Fargate aus verwaltet. Darüber hinaus gibt es eine zugeordnete IAM-Richtlinie, die zum Erteilen von Berechtigungen für Fargate verwendet wird. Bei Aufgaben, die die Fargate-Plattformversion `1.4.0` oder höher verwenden, empfängt die Aufgabe eine einzelne ENI (als Aufgaben-ENI bezeichnet) und der gesamte Netzwerkverkehr fließt durch diese ENI innerhalb Ihrer VPC. Dieser Netzwerkverkehr wird in Ihren VPC-Flow-Protokollen aufgezeichnet. Für Aufgaben, die die Fargate-Plattformversion `1.3.0` und älter verwenden, erhält die Aufgabe zusätzlich zu der Aufgaben-ENI auch eine separate ENI im Besitz von Fargate, die für Netzwerkverkehr verwendet wird, der in den VPC-Flow-Protokollen nicht sichtbar ist. Im der folgenden Tabelle werden das Verhalten des Netzwerkverkehrs sowie die erforderliche IAM-Richtlinie für jede Plattformversion beschrieben.

Aktion	Datenverkehrsfluss mit Linux-Plattformversion **`1.3.0`** und älter	Datenverkehrsfluss mit Linux-Plattformversion **`1.4.0`**	Datenverkehrsfluss mit Windows-Plattformversion **`1.0.0`**	IAM-Berechtigung
Amazon ECR-Anmeldeinformationen abrufen	ENI im Besitz von Fargate	Aufgaben-ENI	Aufgaben-ENI	IAM-Aufgabenausführungsrolle

Aktion	Datenverkehrsfluss mit Linux-Plattformversion **1.3.0** und älter	Datenverkehrsfluss mit Linux-Plattformversion **1.4.0**	Datenverkehrsfluss mit Windows-Plattformversion **1.0.0**	IAM-Berechtigung
Image abrufen	Aufgaben-ENI	Aufgaben-ENI	Aufgaben-ENI	IAM-Aufgabenausführungsrolle
Senden von Protokollen über einen Protokolltreiber	Aufgaben-ENI	Aufgaben-ENI	Aufgaben-ENI	IAM-Aufgabenausführungsrolle
Protokolle FireLens für Amazon ECS senden	Aufgaben-ENI	Aufgaben-ENI	Aufgaben-ENI	IAM-Rolle für Aufgabe
Abrufen von Geheimnissen aus Secrets Manager oder Systems Manager	ENI im Besitz von Fargate	Aufgaben-ENI	Aufgaben-ENI	IAM-Aufgabenausführungsrolle
Datenverkehr des Amazon EFS-Dateisystems	Nicht verfügbar	Aufgaben-ENI	Aufgaben-ENI	IAM-Rolle für Aufgabe
Datenverkehr der Anwendung	Aufgaben-ENI	Aufgaben-ENI	Aufgaben-ENI	IAM-Rolle für Aufgabe

Überlegungen

Beachten Sie Folgendes, wenn Sie Aufgabenvernetzung verwenden.

- Die serviceverknüpfte Amazon ECS-Rolle ist erforderlich, um Amazon ECS die Berechtigungen zu erteilen, in Ihrem Namen Anrufe an andere AWS Services zu tätigen. Diese Rolle wird für Sie erstellt, wenn Sie einen Cluster erstellen oder wenn Sie einen Service in der AWS Management Console erstellen oder aktualisieren. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS. Sie können die serviceverknüpfte Rolle auch mit dem folgenden AWS CLI Befehl erstellen.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Amazon ECS füllt den Hostnamen der Aufgabe mit einem von Amazon bereitgestellten DNS-Hostnamen auf, wenn sowohl die Option `enableDnsHostnames` als auch die Option `enableDnsSupport` auf Ihrer VPC aktiviert sind. Wenn diese Optionen nicht aktiviert sind, ist der DNS-Hostname der Aufgabe ein zufälliger Hostname. Weitere Informationen zu den DNS-Einstellungen für eine ECS finden Sie unter Verwenden von DNS in Ihrer VPC im Amazon-VPC-Benutzerhandbuch.
- Sie können für `awsVpcConfiguration` nur bis zu 16 Subnetze und 5 Sicherheitsgruppen angeben. Weitere Informationen finden Sie unter AwsVpcKonfiguration in der Amazon Elastic Container Service API-Referenz.
- Sie können die von Fargate erstellten und angehängten ENIs nicht manuell trennen oder anhängen. Auf diese Weise wird ein versehentliches Löschen einer ENI, die einer aktuell ausgeführten Aufgabe zugeordnet wird, verhindert. Wenn Sie die ENIs für eine Aufgabe freigeben möchten, stoppen Sie die Aufgabe.
- Wenn ein VPC-Subnetz aktualisiert wird, um die eingestellten DHCP-Optionen zu ändern, können Sie diese Änderungen auch nicht für vorhandene Aufgaben übernehmen, die die VPC verwenden. Starten Sie neue Aufgaben, die die neue Einstellung erhalten, damit die Migration reibungslos funktioniert. Testen Sie die neue Änderung und stoppen Sie dann die alten Aufgaben, wenn kein Rollback erforderlich ist.
- Aufgaben, die in Subnetzen mit IPv6-CIDR-Blöcken gestartet werden, erhalten nur eine IPv6-Adresse, wenn Fargate-Plattformversion `1.4.0` oder höher für Linux oder `1.0.0` für Windows verwendet wird.
- Für Aufgaben, die die Plattformversion `1.4.0` oder höher für Linux oder `1.0.0` für Windows verwenden, unterstützen die Aufgaben-ENIs Jumbo-Frames. Netzwerkschnittstellen sind mit einer Maximum Transmission Unit (MTU) konfiguriert, die der Größe der größten Nutzlast entspricht, die in einen einzelnen Frame passt. Je größer die MTU ist, desto mehr Anwendungsnutzlast passt in ein einzelnes Frame, was den Overhead pro Frame reduziert und die Effizienz erhöht. Die

Unterstützung von Jumbo-Frames reduziert den Overhead, wenn der Netzwerkpfad zwischen Ihrer Aufgabe und dem Ziel Jumbo-Frames unterstützt.

- Services mit Aufgaben, die den Fargate-Starttyp verwenden, unterstützen nur Application Load Balancer oder Network Load Balancer. Classic Load Balancer werden nicht unterstützt. Wenn Sie eine beliebige Zielgruppe für diese Services erstellen, müssen Sie `ip` als Zieltyp auswählen und nicht `instance`. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.

Verwenden einer VPC im Dual-Stack-Modus

Wenn Sie eine VPC im Dual-Stack-Modus verwenden, können Ihre Aufgaben über IPv4, über IPv6 oder über beide kommunizieren. IPv4- und IPv6-Adressen sind voneinander unabhängig und sie müssen daher das Routing und die Sicherheit in Ihrer VPC für IPv4 und IPv6 getrennt konfigurieren. Weitere Informationen zum Konfigurieren der VPC für den Dual-Stack-Modus finden Sie unter Migrieren zu IPv6 im Amazon VPC-Benutzerhandbuch.

Bei Amazon-ECS-Aufgaben auf Fargate wird eine IPv6-Adresse zugewiesen, wenn die folgenden Bedingungen erfüllt sind:

- Ihre Amazon `dualStackIPv6` ECS-Kontoeinstellungen sind für den IAM-Principal aktiviert (`enabled`), der Ihre Aufgaben in der Region startet, in der Sie Ihre Aufgaben starten. Diese Einstellung kann nur mithilfe der API oder AWS CLI geändert werden. Sie haben die Möglichkeit, diese Einstellung für einen bestimmten IAM-Principal in Ihrem Konto oder für Ihr gesamtes Konto zu aktivieren, indem Sie Ihre Kontostandardeinstellung festlegen. Weitere Informationen finden Sie unter Greifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu.
- Ihre VPC und Ihr Subnetz sind für IPv6 aktiviert. Weitere Informationen zum Konfigurieren der VPC für den Dual-Stack-Modus finden Sie unter Migrieren zu IPv6 im Amazon-VPC-Benutzerhandbuch.
- Ihr Subnetz ist für die automatische Zuweisung von IPv6-Adressen aktiviert. Weitere Informationen zur Konfiguration Ihres Subnetzes finden Sie unter Ändern des IPv6-Adressierungsattributs für Ihr Subnetz im Benutzerhandbuch für Amazon VPC.
- Die Aufgabe oder der Service verwendet die Fargate-Plattformversion `1.4.0` oder höher für Linux.

Wenn Sie Ihre VPC mit einem Internet-Gateway oder einem reinen ausgehenden Internet-Gateway konfigurieren, können auf Fargate gehostete Amazon-ECS-Aufgaben, denen eine IPv6-Adresse zugewiesen ist, auf das Internet zugreifen. NAT-Gateways werden nicht benötigt. Weitere

Informationen finden Sie unter Internet-Gateways und Internet-Gateways nur für Egress im Amazon VPC-Benutzerhandbuch.

Speicheroptionen für Amazon ECS-Aufgaben

Amazon ECS bietet Ihnen flexible, kostengünstige und auf Ihre Bedürfnisse zugeschnittene easy-to-use Datenspeicheroptionen. Amazon ECS unterstützt die folgenden Datenvolumenoptionen für Container:

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
Amazon Elastic Block Store (Amazon EBS)	Fargate, Amazon EC2	Linux	Kann beibehalten werden, wenn es an eine eigenständige Aufgabe angehängt wird. Kurzlebig, wenn es an eine Aufgabe angehängt wird, die von einem Dienst verwaltet wird.	Amazon EBS-Volumes bieten kostengünstigen, dauerhaften und leistungsstarken Blockspeicher für datenintensive containerisierte Workloads. Zu den häufigsten Anwendungsfällen gehören transaktionale Workloads wie Datenbanken, virtuelle Desktops und Root-Volumes sowie durchsatzintensive Workloads wie Protokollverarbeitung und ETL-Workloads. Weitere

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
				Informationen finden Sie unter Verwenden Sie Amazon EBS-Volumes mit Amazon ECS.

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
Amazon Elastic File System (Amazon EFS)	Fargate, Amazon EC2	Linux	Persistent	Amazon EFS-Volumes bieten einfachen, skalierbaren und dauerhaften gemeinsamen Dateispeicher für Ihre Amazon ECS-Aufgaben, der automatisch wächst und schrumpft, wenn Sie Dateien hinzufügen und entfernen. Amazon EFS-Volumes unterstützen Parallelität und eignen sich für containerisierte Anwendungen, die horizontal skalieren und Speicherfunktionen wie geringe Latenz, hohen Durchsatz und Konsistenz benötigen. read-after-write Zu den häufigsten Anwendung

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
				sfällen gehören Workloads wie Datenanalyse, Medienverarbeitung, Inhaltsmanagement und Web-Serving. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
Amazon FSx für Windows File Server	Amazon EC2	Windows	Persistent	FSx for Windows File Server Server-Volumes bieten vollständig verwaltete Windows-Dateiserver, mit denen Sie Ihre Windows-Aufgaben bereitstellen können, die persistenten, verteilten, gemeinsam genutzten und statischen Dateispeicher benötigen. Zu den häufigsten Anwendungsfällen zählen.NET-Anwendungen, die möglicherweise lokale Ordner als persistenten Speicher zum Speichern von Anwendungsausgaben benötigen.

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
				Amazon FSx for Windows File Server bietet einen lokalen Ordner im Container, der es mehreren Containern ermöglicht, auf demselben Dateisystem, das von einem SMB Share unterstützt wird, Lese- und Schreibvorgänge durchzuführen. Weitere Informationen finden Sie unter Verwenden Sie FSx for Windows File Server Windows-Dateiserver-Volumes mit Amazon ECS.

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
Docker-Volumes	Amazon EC2	Windows, Linux	Persistent	Docker-Volumes sind eine Funktion der Docker-Container-Laufzeit, die es Containern ermöglicht, Daten dauerhaft zu speichern, indem sie ein Verzeichnis aus dem Dateisystem des Hosts mounten. Docker-Volumetreiber (auch als Plugins bezeichnet) werden verwendet, um Container-Volumes in externe Speichersysteme zu integrieren. Docker-Volumes können über Treiber von Drittanbietern oder über den integrierten Treiber verwaltet werden. `local`

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
				Zu den häufigsten Anwendungsfällen für Docker-Volumes gehören die Bereitstellung persistenter Datenvolumes oder die gemeinsame Nutzung von Volumes an verschiedenen Standorten auf verschiedenen Containern auf derselben Container-Instance. Weitere Informationen finden Sie unter Verwenden Sie Docker-Volumes mit Amazon ECS.

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
Bind-Mounts	Fargate, Amazon EC2	Windows, Linux	Vergänglich	Bind-Mounts bestehen aus einer Datei oder einem Verzeichnis auf dem Host, z. B. einer Amazon EC2 EC2-Instance oder AWS Fargate, das auf einem Container gemountet ist. Zu den häufigsten Anwendungsfällen für Bind-Mounts gehören die gemeinsame Nutzung eines Volumes aus einem Quellcontainer mit anderen Containern in derselben Aufgabe oder das Mounten eines Host-Volumes oder eines leeren Volumes in einem oder mehreren Containern. Weitere Informati

Datenvolumen	Unterstützte Starttypen	Unterstützte Betriebssysteme	Persistenz des Speichers	Anwendungsfälle
				onen finden Sie unter Bind-Mounts mit Amazon ECS verwenden.

Verwenden Sie Amazon EBS-Volumes mit Amazon ECS

Amazon Elastic Block Store (Amazon EBS) -Volumes bieten hochverfügbaren, kostengünstigen, dauerhaften und leistungsstarken Blockspeicher für datenintensive Workloads. Amazon EBS-Volumes können mit Amazon ECS-Aufgaben für Anwendungen mit hohem Durchsatz und transaktionsintensiven Anwendungen verwendet werden.

Während des Starts einer eigenständigen Aufgabe können Sie die Konfiguration angeben, die zum Anhängen eines EBS-Volumes an die Aufgabe verwendet wird. Bei der Erstellung oder Aktualisierung des Dienstes können Sie die Konfiguration angeben, die verwendet wird, um jeder vom ECS-Service verwalteten Aufgabe ein EBS-Volume pro Task zuzuweisen.

Indem Sie die Volume-Konfiguration beim Start und nicht in der Aufgabendefinition angeben, erstellen Sie Aufgabendefinitionen, die nicht auf einen bestimmten Datenvolumetyp oder bestimmte EBS-Volume-Einstellungen beschränkt sind. Anschließend können Sie Ihre Aufgabendefinitionen in verschiedenen Laufzeitumgebungen wiederverwenden. So können Sie beispielsweise während der Bereitstellung für Ihre Produktions-Workloads einen höheren Durchsatz bereitstellen als für Ihre Pre-Prod-Umgebungen.

Amazon EBS-Volumes, die Amazon ECS-Aufgaben zugeordnet sind, werden von Amazon ECS in Ihrem Namen verwaltet. Die Volumes können mit AWS Key Management Service (AWS KMS) -Schlüsseln verschlüsselt werden, um Ihre Daten zu schützen. Sie können entweder neue, leere Volumes für das Anhängen konfigurieren oder Sie können Snapshots verwenden, um Daten aus vorhandenen Volumes zu laden.

Um die Leistung Ihres Volumes zu überwachen, können Sie auch CloudWatch Amazon-Metriken verwenden. Weitere Informationen zu Amazon ECS-Metriken für Amazon EBS-Volumes finden Sie unter Amazon CloudWatch ECS-Metriken und Amazon ECS Container Insights-Metriken.

Weitere Informationen zu Amazon EBS-Volumes finden Sie unter Amazon EBS-Volumes im Amazon EBS-Benutzerhandbuch.

AWS-Regionen und Availability Zones für Amazon EBS-Volumes

Amazon EBS-Volumes können wie folgt AWS-Regionen an Amazon ECS-Aufgaben angehängt werden:

Name der Region	Regionscode
USA Ost (Nord-Virginia)	us-east-1
USA Ost (Ohio)	us-east-2
USA West (Nordkalifornien)	us-west-1
USA West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Melbourne)	ap-southeast-4
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Tokio)	ap-northeast-1
Kanada (Zentral)	ca-central-1
Europa (Frankfurt)	eu-central-1
Europa (Irland)	eu-west-1

Name der Region	Regionscode
Europa (London)	eu-west-2
Europa (Mailand)	eu-south-1
Europa (Paris)	eu-west-3
Europa (Spanien)	eu-south-2
Europa (Stockholm)	eu-north-1
Europa (Zürich)	eu-central-2
Israel (Tel Aviv)	il-central-1
Naher Osten (Bahrain)	me-south-1
Naher Osten (VAE)	me-central-1
Südamerika (São Paulo)	sa-east-1

Important

Sie können Amazon EBS-Volumes nicht für das Anhängen an Fargate Amazon ECS-Aufgaben in den Availability Zones `euc1-az2` und `use1-az3` Availability Zones konfigurieren.

Überlegungen

Beachten Sie bei der Verwendung von Amazon EBS-Volumes Folgendes:

- Amazon EBS-Volumes werden nur für Linux-Aufgaben unterstützt, die auf Fargate gehostet werden, und EC2-Startaufgaben, die auf Nitro basierten Linux-Instances mit Amazon ECS-optimierten Amazon Machine Images (AMIs) gehostet werden. Weitere Informationen zu Instance-Typen finden Sie unter Instance-Typen im Amazon EC2 EC2-Benutzerhandbuch. Weitere Informationen zu Amazon ECS-Starttypen finden Sie unterAmazon-ECS-Starttypen.

- Für Aufgaben, die auf Fargate gehostet werden, werden Amazon EBS-Volumes auf der Plattformversion `1.4.0` oder höher (Linux) unterstützt. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
- Für Aufgaben, die auf Amazon EC2 EC2-Linux-Instances gehostet werden, werden Amazon EBS-Volumes auf ECS-optimiertem AMI oder höher unterstützt. `20231219` Weitere Informationen finden Sie unter Abrufen von Amazon-ECS-optimierten AMI-Metadaten.
- Der Amazon EBS-Volumetyp magnetisch (`standard`) wird für Aufgaben, die auf Fargate gehostet werden, nicht unterstützt. Weitere Informationen zu Amazon EBS-Volumetypen finden Sie unter Amazon EBS-Volumes im Amazon EC2 EC2-Benutzerhandbuch.
- Eine IAM-Rolle für die Amazon ECS-Infrastruktur ist erforderlich, wenn Sie einen Service oder eine eigenständige Aufgabe erstellen, bei der ein Volume bei der Bereitstellung konfiguriert wird. Sie können die AWS verwaltete `AmazonECSInfrastructureRolePolicyForVolumes` IAM-Richtlinie an die Rolle anhängen, oder Sie können die verwaltete Richtlinie als Leitfaden verwenden, um Ihre eigene Richtlinie mit Berechtigungen zu erstellen und anzuhängen, die Ihren spezifischen Anforderungen entsprechen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon ECS-Infrastruktur.
- Sie können jeder Amazon ECS-Aufgabe höchstens ein Amazon EBS-Volume zuordnen, und es muss sich dabei um ein neues Volume handeln. Sie können kein vorhandenes Amazon EBS-Volume an eine Aufgabe anhängen. Sie können jedoch bei der Bereitstellung ein neues Amazon EBS-Volume konfigurieren, indem Sie den Snapshot eines vorhandenen Volumes verwenden.
- Sie können Amazon EBS-Volumes bei der Bereitstellung nur für Services konfigurieren, die den Bereitstellungstyp für fortlaufende Updates und die Replica-Planungsstrategie verwenden.
- Amazon ECS fügt automatisch die reservierten Tags `AmazonECSCreated` und dem `AmazonECSManaged` angehängten Volume hinzu. Wenn Sie diese Tags aus dem Volume entfernen, kann Amazon ECS das Volume nicht in Ihrem Namen verwalten. Weitere Informationen zum Taggen von Amazon EBS-Volumes finden Sie unter Tagging Amazon EBS-Volumes. Weitere Informationen zum Taggen von Amazon ECS-Ressourcen finden Sie unter Taggen Ihrer Amazon ECS-Ressourcen.
- Die Bereitstellung von Volumes aus einem Snapshot eines Amazon EBS-Volumes, das Partitionen enthält, wird nicht unterstützt.
- Volumes, die Aufgaben zugeordnet sind, die von einem Service verwaltet werden, bleiben nicht erhalten und werden bei Beendigung der Aufgabe immer gelöscht.
- Sie können Amazon EBS-Volumes nicht so konfigurieren, dass sie an Amazon ECS-Aufgaben angehängt werden, die auf AWS Outposts ausgeführt werden.

Verschieben Sie die Volume-Konfiguration in einer Amazon ECS-Aufgabendefinition auf die Startzeit

Um ein Amazon EBS-Volume für das Anhängen an Ihre Aufgabe zu konfigurieren, müssen Sie die Mount-Point-Konfiguration in Ihrer Aufgabendefinition angeben und das Volume benennen. Sie müssen auch `configuredAtLaunch` auf einstellen, `true` da Amazon EBS-Volumes in der Aufgabendefinition nicht für das Anhängen konfiguriert werden können. Stattdessen werden Amazon EBS-Volumes so konfiguriert, dass sie während der Bereitstellung angehängt werden.

Die folgende Aufgabendefinition zeigt die Syntax für die `volumes` Objekte `mountPoints` und in der Aufgabendefinition. Weitere Informationen zu Aufgabendefinitionsparametern finden Sie unterAmazon ECS-Aufgabendefinitionsparameter. Wenn Sie dieses Beispiel verwenden möchten, ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

Um die Aufgabendefinition mithilfe von AWS Command Line Interface (AWS CLI) zu registrieren, speichern Sie die Vorlage als JSON-Datei und übergeben Sie die Datei dann als Eingabe für den `register-task-definition` Befehl.

Informationen zum Erstellen und Registrieren einer Aufgabendefinition mithilfe von finden Sie unterErstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole. AWS Management Console

```
{
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
```

```
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Typ: Objekt-Array

Erforderlich: Nein

Die Bereitstellungspunkte für die Datenvolumes in Ihrem Container. Dieser Parameter ordnet zu `Volumes` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volume` für die docker run zu.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten. Windows-Container können keine Verzeichnisse auf einem anderen Laufwerk mounten, und Bereitstellungspunkte können nicht laufwerksübergreifend verwendet werden. Sie müssen Bereitstellungspunkte angeben, um ein Amazon EBS-Volume direkt an eine Amazon ECS-Aufgabe anzuhängen.

`sourceVolume`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Name des zu mountenden Volumes.

`containerPath`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Pfad im Container, in dem das Volume bereitgestellt werden soll.

`readOnly`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume. Der Standardwert ist `false`.

`name`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Volumes. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Zahlen, Bindestriche () und Unterstriche (`-`) sind zulässig. _ Auf diesen Namen wird im `sourceVolume` Parameter des Container-Definitionsobjekts verwiesen. `mountPoints`

`configuredAtLaunch`

Typ: Boolesch

Erforderlich: Ja, wenn Sie das direkte Anhängen eines EBS-Volumes an eine Aufgabe verwenden möchten.

Gibt an, ob ein Volume beim Start konfigurierbar ist. Wenn diese Option auf gesetzt ist`true`, können Sie das Volume konfigurieren, wenn Sie eine eigenständige Aufgabe ausführen oder wenn Sie einen Dienst erstellen oder aktualisieren. Wenn diese Option auf gesetzt ist`true`, können Sie in der Aufgabendefinition keine andere Volume-Konfiguration angeben. Dieser Parameter muss bereitgestellt und auf gesetzt werden, `true` um ein Amazon EBS-Volume für das Anhängen an eine Aufgabe zu konfigurieren.

Verschlüsseln Sie auf Amazon EBS-Volumes gespeicherte Daten für Amazon ECS

Sie können AWS Key Management Service (AWS KMS) verwenden, um kryptografische Schlüssel zu erstellen und zu verwalten, die Ihre Daten schützen. Amazon EBS-Volumes werden im Ruhezustand mithilfe AWS KMS keys von verschlüsselt. Die folgenden Datentypen werden verschlüsselt:

- Daten, die im Ruhezustand auf dem Volume gespeichert sind
- Festplatten-I/O
- Aus dem Volume erstellte Snapshots
- Neue Volumes, die aus Snapshots erstellt wurden

Sie können die Amazon EBS-Verschlüsselung standardmäßig so konfigurieren, dass alle neu erstellten und an eine Aufgabe angehängten Volumes mithilfe des KMS-Schlüssels, den Sie für Ihr Konto konfigurieren, verschlüsselt werden. Weitere Informationen zur Amazon EBS-Verschlüsselung und Standardverschlüsselung finden Sie unter Amazon EBS-Verschlüsselung im Amazon EC2 EC2-Benutzerhandbuch.

Amazon EBS-Volumes, die an Aufgaben angehängt sind, können entweder mithilfe eines Standardschlüssels Von AWS verwalteter Schlüssel mit dem Alias oder eines symmetrischen`alias/aws/ebs`, vom Kunden verwalteten Schlüssels verschlüsselt werden. Standardwerte Von AWS verwaltete Schlüssel sind für jeden Benutzer einzigartig AWS-Konto AWS-Region und werden automatisch erstellt. Um einen symmetrischen, vom Kunden verwalteten Schlüssel zu erstellen, folgen Sie den Schritten unter KMS-Schlüssel für symmetrische Verschlüsselung im AWS KMS Entwicklerhandbuch.

Richtlinie für vom Kunden verwaltete KMS-Schlüssel

Um ein EBS-Volume, das an Ihre Aufgabe angehängt ist, mithilfe Ihres vom Kunden verwalteten Schlüssels zu verschlüsseln, müssen Sie Ihre KMS-Schlüsselrichtlinie so konfigurieren, dass die IAM-Rolle, die Sie für die Volume-Konfiguration verwenden, über die erforderlichen Berechtigungen zur Verwendung des Schlüssels verfügt. Die Schlüsselrichtlinie muss die Berechtigungen und enthalten. `kms:CreateGrant` `kms:GenerateDataKey*` Die `kms:ReEncryptFrom` Berechtigungen `kms:ReEncryptTo` und sind für die Verschlüsselung von Volumes erforderlich, die mithilfe von Snapshots erstellt wurden. Wenn Sie nur neue, leere Volumes für das Anhängen konfigurieren und verschlüsseln möchten, können Sie die `kms:ReEncryptTo` Berechtigungen und ausschließen. `kms:ReEncryptFrom`

Der folgende JSON-Snippet zeigt wichtige Richtlinienerklärungen, die Sie an Ihre KMS-Schlüsselrichtlinie anhängen können. Durch die Verwendung dieser Anweisungen erhält ECS Zugriff auf die Verwendung des Schlüssels für die Verschlüsselung des EBS-Volumes. Wenn Sie die Beispielrichtlinien verwenden möchten, ersetzen Sie sie durch Ihre eigenen Informationen. *user input placeholders* Konfigurieren Sie wie immer nur die Berechtigungen, die Sie benötigen.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
```

```
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

Weitere Informationen zu wichtigen Richtlinien und Berechtigungen finden Sie unter Wichtige Richtlinien AWS KMS und AWS KMS Berechtigungen im AWS KMS Entwicklerhandbuch. Informationen zur Behebung von Problemen mit dem Anhängen von EBS-Volumes im Zusammenhang mit Schlüsselberechtigungen finden Sie unterFehlerbehebung bei Amazon EBS-Volume-Anhängen an Amazon ECS-Aufgaben .

Geben Sie die Amazon EBS-Volume-Konfiguration bei der Amazon ECS-Bereitstellung an

Nachdem Sie eine Aufgabendefinition mit dem `configuredAtLaunch` Parameter auf registriert haben`true`, können Sie ein Amazon EBS-Volume bei der Bereitstellung konfigurieren, wenn Sie eine eigenständige Aufgabe ausführen oder wenn Sie einen Service erstellen oder aktualisieren.

Um ein Volume zu konfigurieren, können Sie die Amazon ECS-APIs verwenden oder eine JSON-Datei als Eingabe für die folgenden AWS CLI Befehle übergeben:

- `run-task`um eine eigenständige ECS-Aufgabe auszuführen.
- `start-task`um eine eigenständige ECS-Aufgabe in einer bestimmten Container-Instance auszuführen. Dieser Befehl gilt nicht für Aufgaben vom Typ Fargate Launch.
- `create-service`um einen neuen ECS-Service zu erstellen.
- `update-service`um einen bestehenden Dienst zu aktualisieren.

> **Note**
>
> Damit ein Container in Ihrer Aufgabe auf das bereitgestellte Amazon EBS-Volume schreiben kann, müssen Sie den Container als Root-Benutzer ausführen.

Sie können ein Amazon EBS-Volume auch mit dem AWS Management Console konfigurieren. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen, Einen Amazon ECS-Service mithilfe der Konsole erstellen und Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.

Der folgende JSON-Snippet zeigt alle Parameter eines Amazon EBS-Volumes, die bei der Bereitstellung konfiguriert werden können. Um diese Parameter für die Volume-Konfiguration zu verwenden, ersetzen Sie sie durch Ihre *user input placeholders* eigenen Informationen. Weitere Informationen zu diesen Parametern finden Sie unter Volume-Konfigurationen.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "arn:aws::iam:1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                },
                "filesystemType": "ext4"
            }
        }
    ]
```

> **Important**
>
> Stellen `volumeName` Sie sicher, dass die, die `volumeName` Sie in der Konfiguration angeben, mit der in Ihrer Aufgabendefinition angegebenen übereinstimmt.

Informationen zur Überprüfung des Status eines Volume-Anhangs finden Sie unterFehlerbehebung bei Amazon EBS-Volume-Anhängen an Amazon ECS-Aufgaben . Informationen zur Amazon ECS-Infrastrukturrolle AWS Identity and Access Management (IAM), die für das Anhängen von EBS-Volumes erforderlich ist, finden Sie unter. IAM-Rolle für die Amazon ECS-Infrastruktur

Im Folgenden finden Sie Beispiele für JSON-Snippets, die die Konfiguration von Amazon EBS-Volumes zeigen. Diese Beispiele können verwendet werden, indem die Snippets in JSON-Dateien gespeichert und die Dateien als Parameter (unter Verwendung des `--cli-input-json file://filename` Parameters) für Befehle übergeben werden. AWS CLI Ersetzen Sie *`user input placeholders`* durch Ihre Informationen.

Konfigurieren Sie ein Volume für eine eigenständige Aufgabe

Der folgende Ausschnitt zeigt die Syntax für die Konfiguration von Amazon EBS-Volumes für den Anhang an eine eigenständige Aufgabe. Der folgende JSON-Snippet zeigt die Syntax für die Konfiguration der Einstellungen`volumeType`,, und`sizeInGiB`. `encrypted` `kmsKeyId` Die in der JSON-Datei angegebene Konfiguration wird verwendet, um ein EBS-Volume zu erstellen und an die eigenständige Aufgabe anzuhängen.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
```

```
    ]
}
```

Konfigurieren Sie ein Volume bei der Diensterstellung

Der folgende Ausschnitt zeigt die Syntax für die Konfiguration von Amazon EBS-Volumes für den Anhang an Aufgaben, die von einem Service verwaltet werden. Die Volumes werden mithilfe von aus dem Snapshot bezogen. `snapshotId` Die in der JSON-Datei angegebene Konfiguration wird verwendet, um ein EBS-Volume zu erstellen und an jede vom Service verwaltete Aufgabe anzuhängen.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345"
            }
        }
   ]
}
```

Konfigurieren Sie ein Volume bei der Service-Aktualisierung

Der folgende JSON-Snippet zeigt die Syntax für die Aktualisierung eines Services, für den zuvor keine Amazon EBS-Volumes für das Anhängen an Aufgaben konfiguriert waren. Sie müssen den ARN einer Aufgabendefinitionsrevision mit der `configuredAtLaunch` Einstellung auf angeben`true`. Der folgende JSON-Snippet zeigt die Syntax für die Konfiguration der Einstellungen `volumeType` `sizeInGiB``throughput`,`iops`, und`filesystemType`. Diese Konfiguration wird verwendet, um ein EBS-Volume zu erstellen und an jede vom Service verwaltete Aufgabe anzuhängen.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
```

```
   "serviceName": "mysvc",
   "desiredCount": 2,
  "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

Einen Service so konfigurieren, dass er Amazon EBS-Volumes nicht mehr nutzt

Der folgende JSON-Snippet zeigt die Syntax für die Aktualisierung eines Service, sodass er keine Amazon EBS-Volumes mehr verwendet. Sie müssen den ARN einer Aufgabendefinition mit `configuredAtLaunch` set to `false` oder einer Aufgabendefinition ohne den `configuredAtLaunch` Parameter angeben. Sie müssen auch ein leeres `volumeConfigurations` Objekt angeben.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": []
}
```

Kündigungsrichtlinie für Amazon EBS-Volumes

Wenn eine Amazon ECS-Aufgabe beendet wird, bestimmt Amazon ECS anhand des `deleteOnTermination` Werts, ob das Amazon EBS-Volume, das mit der beendeten Aufgabe verknüpft ist, gelöscht werden soll. Standardmäßig werden EBS-Volumes, die Aufgaben zugeordnet sind, gelöscht, wenn die Aufgabe beendet wird. Bei eigenständigen Aufgaben können Sie diese Einstellung ändern, sodass das Volume beim Beenden der Aufgabe erhalten bleibt.

> **Note**
>
> Volumes, die an Aufgaben angehängt sind, die von einem Dienst verwaltet werden, bleiben nicht erhalten und werden immer gelöscht, wenn die Aufgabe beendet wird.

Schlagwort: Amazon EBS-Volumen

Sie können Amazon EBS-Volumes mithilfe des `tagSpecifications` Objekts taggen. Mithilfe des Objekts können Sie Ihre eigenen Tags angeben und die Weitergabe von Tags anhand der Aufgabendefinition oder des Service festlegen, je nachdem, ob das Volume an eine eigenständige Aufgabe oder eine Aufgabe in einem Service angehängt ist. Die maximale Anzahl von Tags, die an ein Volume angehängt werden können, ist 50.

> **Important**
>
> Amazon ECS hängt die Tags `AmazonECSCreated` und die `AmazonECSManaged` reservierten Tags automatisch an ein Amazon EBS-Volume an. Das bedeutet, dass Sie das Anhängen von maximal 48 zusätzlichen Tags an ein Volume steuern können. Bei diesen zusätzlichen Tags kann es sich um benutzerdefinierte, von ECS verwaltete oder weitergegebene Tags handeln.

Wenn Sie Ihrem Volume von Amazon ECS verwaltete Tags hinzufügen möchten, müssen Sie `true` in Ihrem`UpdateService`,`CreateService`, `RunTask` Anruf `enableECSManagedTags` auf einstellen. `StartTask` Wenn Sie von Amazon ECS verwaltete Tags aktivieren, markiert Amazon ECS das Volume automatisch mit Cluster- und Serviceinformationen (`aws:ecs:`*`clusterName`*und`aws:ecs:`*`serviceName`*). Weitere Informationen zum Taggen von Amazon ECS-Ressourcen finden Sie unter Taggen Ihrer Amazon ECS-Ressourcen.

Der folgende JSON-Snippet zeigt die Syntax für die Kennzeichnung jedes Amazon EBS-Volumes, das an jede Aufgabe in einem Service angehängt ist, mit einem benutzerdefinierten Tag. Um dieses Beispiel für die Erstellung eines Service zu verwenden, ersetzen Sie das *`user input placeholders`* durch Ihre eigenen Informationen.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
```

```
    "serviceName": "mysvc",
    "desiredCount": 2,
    "enableECSManagedTags": true,
    "volumeConfigurations": [
         {
             "name": "datadir",
             "managedEBSVolume": {
                 "volumeType": "gp3",
                 "sizeInGiB": 100,
                  "tagSpecifications": [
                     {
                         "resourceType": "volume",
                         "tags": [
                             {
                                 "key": "key1",
                                 "value": "value1"
                             }
                         ],
                         "propagateTags": "NONE"
                     }
                 ]
                 "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                 "encrypted": true,
                 "kmsKeyId":
  "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
             }
         }
    ]
}
```

> **Important**
>
> Sie müssen einen `volume` Ressourcentyp angeben, um Amazon EBS-Volumes zu kennzeichnen.

Leistung von Amazon EBS-Volumes für Fargate-On-Demand-Aufgaben

Das für eine Fargate-On-Demand-Aufgabe verfügbare Basis-Volume (IOPS) und der Durchsatz hängen von der Gesamtzahl der CPU-Einheiten ab, die Sie für die Aufgabe anfordern. Wenn Sie 0,25, 0,5 oder 1 virtuelle CPU-Einheit (vCPU) für Ihre Fargate-Aufgabe anfordern, empfehlen wir Ihnen, ein Allzweck-SSD-Volume (gp2odergp3) oder ein Festplattenlaufwerk (HDD) -Volume

(`st1`oder) zu konfigurieren. `sc1` Wenn Sie mehr als 1 vCPU für Ihre Fargate-Aufgabe anfordern, gelten die folgenden grundlegenden Leistungsgrenzen für ein Amazon EBS-Volume, das der Aufgabe zugeordnet ist. Möglicherweise erhalten Sie vorübergehend eine höhere EBS-Leistung als die folgenden Grenzwerte. Wir empfehlen Ihnen jedoch, Ihre Arbeitslast auf der Grundlage dieser Grenzwerte zu planen.

Angeforderte CPU-Einheiten (in vCPUs)	Basiswerte Amazon EBS IOPS (16 KiB I/O)	Amazon EBS-Basis durchsatz (in MiBps, 128 KiB I/O)	Basisbandbreite (in Mbit/s)
2	3,000	75	360
4	5,000	120	1.150
8	10.000	250	2.300
16	15 000	500	4.500

> **Note**
>
> Wenn Sie ein Amazon EBS-Volume für das Anhängen an eine Fargate-Aufgabe konfigurieren, wird das Amazon EBS-Leistungslimit für die Fargate-Aufgabe zwischen dem kurzlebigen Speicher der Aufgabe und dem angehängten Volume gemeinsam genutzt.

Fehlerbehebung bei Amazon EBS-Volume-Anhängen an Amazon ECS-Aufgaben

Möglicherweise müssen Sie das Anhängen von Amazon EBS-Volumes an Amazon ECS-Aufgaben beheben oder überprüfen.

Überprüfen Sie den Status des Anhangs des Volum

Sie können den verwenden AWS Management Console , um den Status des Anhangs eines Amazon EBS-Volumes an eine Amazon ECS-Aufgabe anzuzeigen. Wenn die Aufgabe gestartet wird und der Anhang fehlschlägt, wird Ihnen auch ein Statusgrund angezeigt, den Sie zur Fehlerbehebung verwenden können. Das erstellte Volume wird gelöscht und die Aufgabe wird gestoppt. Weitere Hinweise zu Statusgründen finden Sie unterStatusgründe für die Zuordnung von Amazon EBS-Volumes zu Amazon ECS-Aufgaben.

Um den Anhangsstatus und den Grund für den Status eines Volumes in der Konsole einzusehen

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Seite Cluster den Cluster aus, in dem Ihre Aufgabe ausgeführt wird. Die Detailseite des Clusters wird angezeigt.
3. Wählen Sie auf der Detailseite des Clusters die Registerkarte Aufgaben aus.
4. Wählen Sie die Aufgabe aus, für die Sie den Status des Volume-Anhangs anzeigen möchten. Wenn die Aufgabe, die Sie untersuchen möchten, beendet wurde, müssen Sie möglicherweise „Nach gewünschtem Status filtern“ auswählen und „Gestoppt“ wählen.
5. Wählen Sie auf der Detailseite der Aufgabe die Registerkarte Volumes aus. Sie können den Anhangsstatus des Amazon EBS-Volumes unter Anhangsstatus sehen. Wenn das Volume nicht an die Aufgabe angehängt werden kann, können Sie den Status unter Status des Anhangs auswählen, um die Ursache des Fehlers anzuzeigen.

Mithilfe der DescribeTasksAPI können Sie auch den Status des Volumens des Anhangs einer Aufgabe und den Grund für den zugehörigen Status anzeigen.

Fehler bei Diensten und Aufgaben

Möglicherweise treten Service- oder Taskfehler auf, die nicht spezifisch für Amazon EBS-Volumes sind und sich auf die Volume-Zuordnung auswirken können. Weitere Informationen finden Sie unter

- Service-Ereignismeldungen
- Fehlercodes für gestoppte Aufgaben
- Gründe für API-Fehler

Statusgründe für die Zuordnung von Amazon EBS-Volumes zu Amazon ECS-Aufgaben

Verwenden Sie die folgende Referenz, um Probleme zu beheben, die in Form von Statusgründen AWS Management Console bei der Konfiguration von Amazon EBS-Volumes für das Anhängen an Amazon ECS-Aufgaben auftreten können. Weitere Informationen zum Auffinden dieser Statusgründe in der Konsole finden Sie unterÜberprüfen Sie den Status des Anhangs des Volum.

ECS konnte die konfigurierte ECS-Infrastrukturrolle 'arn:aws:iam:: 111122223333:role/ ecs 'nicht annehmen. InfrastructureRole Bitte stellen Sie sicher, dass für die übergebene Rolle das richtige Vertrauensverhältnis zu Amazon ECS besteht

Dieser Statusgrund tritt in den folgenden Szenarien auf.

- Sie geben eine IAM-Rolle an, ohne dass die erforderliche Vertrauensrichtlinie beigefügt ist. Amazon ECS kann nicht auf die von Ihnen angegebene Amazon ECS-Infrastruktur-IAM-Rolle zugreifen, wenn die Rolle nicht über die erforderliche Vertrauensrichtlinie verfügt. Die Aufgabe kann im `DEPROVISIONING` Status hängen bleiben. Weitere Informationen zu den erforderlichen Vertrauensrichtlinien finden Sie unterIAM-Rolle für die Amazon ECS-Infrastruktur.
- Ihr IAM-Benutzer ist nicht berechtigt, die Amazon ECS-Infrastrukturrolle an Amazon ECS weiterzugeben. Die Aufgabe kann im `DEPROVISIONING` Status hängen bleiben. Um dieses Problem zu vermeiden, können Sie die `PassRole` Erlaubnis an Ihren Benutzer anhängen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon ECS-Infrastruktur.
- Ihre IAM-Rolle verfügt nicht über die erforderlichen Berechtigungen für das Anhängen von Amazon EBS-Volumes. Die Aufgabe kann im Status hängen bleiben. `DEPROVISIONING` Weitere Informationen zu den spezifischen Berechtigungen, die für das Anhängen von Amazon EBS-Volumes an Aufgaben erforderlich sind, finden Sie unter. IAM-Rolle für die Amazon ECS-Infrastruktur

> **Note**
>
> Möglicherweise wird Ihnen diese Fehlermeldung auch aufgrund einer Verzögerung bei der Rollenübertragung angezeigt. Wenn der erneute Versuch, die Rolle nach einigen Minuten zu verwenden, das Problem nicht behebt, haben Sie möglicherweise die Vertrauensrichtlinie für die Rolle falsch konfiguriert.

ECS konnte das EBS-Volume nicht einrichten. Es wurde festgestellt IdempotentParameterMismatch „; „Das von Ihnen angegebene Client-Token ist mit einer Ressource verknüpft, die bereits gelöscht wurde. Bitte verwenden Sie ein anderes Client-Token.“

Die folgenden AWS KMS wichtigen Szenarien können dazu führen, dass eine `IdempotentParameterMismatch` Meldung erscheint:

- Sie geben einen KMS-Schlüssel-ARN, eine ID oder einen Alias an, der nicht gültig ist. In diesem Szenario scheint die Aufgabe erfolgreich gestartet zu werden, aber die Aufgabe schlägt irgendwann fehl, weil der KMS-Schlüssel asynchron AWS authentifiziert wird.

Weitere Informationen finden Sie unter Amazon EBS-Verschlüsselung im Amazon EC2 EC2-Benutzerhandbuch.

- Sie stellen einen vom Kunden verwalteten Schlüssel bereit, dem die Berechtigungen fehlen, die es der IAM-Rolle der Amazon ECS-Infrastruktur ermöglichen, den Schlüssel für die Verschlüsselung zu verwenden. Informationen zur Vermeidung von Problemen mit Schlüsselrichtlinien finden Sie in der AWS KMS Beispielschlüsselrichtlinie unter Datenverschlüsselung für Amazon EBS-Volumes.

Sie können Amazon so einrichten EventBridge , dass Amazon EBS-Volumenereignisse und Amazon ECS-Ereignisse zur Änderung des Aufgabenstatus an ein Ziel, z. B. CloudWatch Amazon-Gruppen, gesendet werden. Sie können diese Ereignisse dann verwenden, um das spezifische vom Kunden verwaltete Schlüsselproblem zu identifizieren, das sich auf die Volumenzuweisung ausgewirkt hat. Weitere Informationen finden Sie unter

- Wie kann ich eine CloudWatch Protokollgruppe erstellen, die als Ziel für eine EventBridge Regel verwendet werden soll? auf AWS re:POST.
- Ereignisse zur Änderung des Aufgabenstatus.
- EventBridgefür Amazon EBS im Amazon EBS-Benutzerhandbuch.

ECS hat bei der Konfiguration des EBS-Volume-Anhangs zu Ihrer Task eine Zeitüberschreitung erlitten.

Die folgenden Szenarien im Dateisystemformat führen zu dieser Meldung.

- Das Dateisystemformat, das Sie bei der Konfiguration angeben, ist nicht mit dem Betriebssystem der Aufgabe kompatibel.
- Sie konfigurieren ein Amazon EBS-Volume so, dass es aus einem Snapshot erstellt wird, und das Dateisystemformat des Snapshots ist nicht mit dem Betriebssystem der Aufgabe kompatibel. Für Volumes, die aus einem Snapshot erstellt wurden, müssen Sie denselben Dateisystemtyp angeben, den das Volume bei der Erstellung des Snapshots verwendet hat.

Sie können die Amazon ECS-Container-Agent-Protokolle verwenden, um diese Meldung für Amazon EC2 EC2-Startaufgaben zu beheben. Weitere Informationen finden Sie unter Speicherorte von Amazon ECS-Protokolldateien und Amazon ECS-Protokollsammler.

Verwenden Sie Amazon EFS-Volumes mit Amazon ECS

Amazon Elastic File System (Amazon EFS) stellt einen einfachen, skalierbaren Dateispeicher für die Verwendung mit Ihren Amazon-ECS-Aufgaben bereit. Mit Amazon EFS ist die Speicherkapazität

elastisch. Sie wird automatisch erweitert oder verkleinert, wenn Sie Dateien hinzufügen oder entfernen. Ihre Anwendungen verfügen jederzeit über den jeweils erforderlichen Speicherplatz.

Sie können Amazon EFS-Dateisysteme mit Amazon ECS verwenden, um Dateisystemdaten in Ihre gesamte Flotte von Container-Instances zu exportieren. Auf diese Weise haben alle Ihre Aufgaben Zugriff auf denselben persistenten Speicher, unabhängig von der Instance, auf der sie landen. Ihre Aufgabendefinitionen müssen auf Volumen-Mounts auf der Container-Instance verweisen, um das Dateisystem verwenden zu können.

Ein Tutorial finden Sie unter Konfiguration von Amazon EFS-Dateisystemen für Amazon ECS mithilfe der Konsole.

Überlegungen

Bei der Verwendung von Amazon-EFS-Volumes sollte Folgendes berücksichtigt werden:

- Bei Aufgaben, die den EC2-Starttyp verwenden, wurde die Amazon-EFS-Dateisystemunterstützung als öffentliche Vorschau mit Amazon-ECS-optimierter AMI-Version `20191212` mit Container-Agent Version 1.35.0 hinzugefügt. Die Amazon-EFS-Dateisystemunterstützung erreichte jedoch die allgemeine Verfügbarkeit mit Amazon-ECS-optimierter AMI-Version `20200319` mit Containeragent Version 1.38.0, die die Amazon-EFS-Zugangspunkt- und IAM-Autorisierungsfeatures enthielt. Wir empfehlen die Verwendung der Amazon-ECS-optimierten AMI-Version `20200319` oder höher, um diese Features zu nutzen. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

  > **Note**
  >
  > Wenn Sie Ihr eigenes AMI erstellen, müssen Sie den Containeragent 1.38.0 oder höher, `ecs-init`-Version 1.38.0-1 oder höher verwenden und die folgenden Befehle für Ihre Amazon-EC2-Instance ausführen, um das Amazon-ECS-Volume-Plugin zu aktivieren. Die Befehle hängen davon ab, ob Sie Amazon Linux 2 oder Amazon Linux als Basis-Image verwenden.
  >
  > Amazon Linux 2
  >
  > ```
  > yum install amazon-efs-utils
  > systemctl enable --now amazon-ecs-volume-plugin
  > ```
  >
  > Amazon Linux

```
yum install amazon-efs-utils
sudo shutdown -r now
```

- Bei Aufgaben, die auf Fargate gehostet werden, werden Amazon-EFS-Dateisysteme von Plattformversion 1.4.0 oder höher (Linux) unterstützt. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
- Bei der Verwendung von Amazon EFS Volumes für Aufgaben, die auf Fargate gehostet werden, erstellt Fargate einen Supervisor-Container erstellt, der für die Verwaltung des Amazon EFS Volumes zuständig ist. Der Supervisor-Container beansprucht nur eine kleine Menge des Arbeitsspeichers der Aufgabe. Der Supervisor-Container ist beim Abfragen des Endpunkts der Aufgabenmetadaten Version 4 sichtbar. Darüber hinaus ist er in CloudWatch Container Insights als Containername sichtbar`aws-fargate-supervisor`. Weitere Informationen zur Verwendung des Amazon EC2 EC2-Starttyps finden Sie unterAmazon ECS-Endpunkt für Aufgabenmetadaten, Version 4. Weitere Informationen zur Verwendung des Starttyps Fargate finden Sie unterAmazon ECS-Endpunkt für Aufgabenmetadaten, Version 4, für Aufgaben auf Fargate.
- Verwenden von Amazon-EFS-Volumes oder Angeben eines `EFSVolumeConfiguration` wird auf externen Instances nicht unterstützt.
- Es wird empfohlen, den `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`-Parameter in der Agentenkonfigurationsdatei auf einen Wert festzulegen, der kleiner als der Standardwert ist (ca. 1 Stunde). Diese Änderung verhindert, dass die EFS-Mount-Anmeldeinformationen ablaufen, und ermöglicht die Bereinigung von Mounts, die nicht verwendet werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Verwenden Sie Amazon EFS-Zugriffspunkte

Amazon EFS Access Points sind anwendungsspezifische Einstiegspunkte in ein EFS-Dateisystem, um den Anwendungszugriff auf freigegebene Datensätze zu verwalten. Weitere Informationen zu Amazon EFS Access Points und zur Steuerung des Zugriffs auf diese finden Sie unter Arbeiten mit Amazon EFS Access Points im Amazon Elastic File System-Benutzerhandbuch.

Zugriffspunkte können eine Benutzeridentität, einschließlich der POSIX-Gruppen des Benutzers, für alle Dateisystemanforderungen erzwingen, die über den Zugriffspunkt erfolgen. Zugriffspunkte können auch ein anderes Stammverzeichnis für das Dateisystem erzwingen. Auf diese Weise können Clients nur auf Daten im angegebenen Verzeichnis oder in den dazugehörigen Unterverzeichnissen zugreifen.

> **Note**
>
> Beim Erstellen eines EFS-Zugriffspunkts geben Sie einen Pfad im Dateisystem an, der als Stammverzeichnis dienen soll. Wenn Sie auf das EFS-Dateisystem mit einer Zugriffspunkt-ID in Ihrer Amazon-ECS-Aufgabendefinition verweisen, muss das Stammverzeichnis ausgelassen oder auf `/` festgelegt sein. Das erzwingt den auf dem EFS Zugriffspunkt festgelegten Pfad.

Mithilfe einer Amazon-ECS-Aufgaben-IAM-Rolle können Sie erzwingen, dass bestimmte Anwendungen einen bestimmten Zugriffspunkt verwenden. Durch die Kombination von IAM-Richtlinien mit Zugriffspunkten können Sie einen sicheren Zugriff auf bestimmte Datensätze für Ihre Anwendungen bereitstellen. Weitere Informationen zum Verwenden von Aufgaben-IAM-Rollen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

Bewährte Methoden für die Verwendung von Amazon EFS-Volumes mit Amazon ECS

Beachten Sie die folgenden Best-Practice-Empfehlungen, wenn Sie Amazon EFS mit Amazon ECS verwenden.

Sicherheits- und Zugriffskontrollen für Amazon EFS-Volumes

Amazon EFS bietet Funktionen zur Zugriffskontrolle, mit denen Sie sicherstellen können, dass die in einem Amazon EFS-Dateisystem gespeicherten Daten sicher sind und nur für Anwendungen zugänglich sind, die sie benötigen. Sie können Daten schützen, indem Sie die Verschlüsselung im Ruhezustand und bei der Übertragung aktivieren. Weitere Informationen finden Sie unter Datenverschlüsselung in Amazon EFS im Benutzerhandbuch zu Amazon Elastic File System.

Neben der Datenverschlüsselung können Sie Amazon EFS auch verwenden, um den Zugriff auf ein Dateisystem einzuschränken. Es gibt drei Möglichkeiten, die Zugriffskontrolle in EFS zu implementieren.

- Sicherheitsgruppen — Mit Amazon EFS-Mount-Zielen können Sie eine Sicherheitsgruppe konfigurieren, die verwendet wird, um Netzwerkverkehr zuzulassen und abzulehnen. Sie können die an Amazon EFS angehängte Sicherheitsgruppe so konfigurieren, dass NFS-Verkehr (Port 2049) von der Sicherheitsgruppe zugelassen wird, die Ihren Amazon ECS-Instances zugeordnet ist, oder, wenn Sie den `awsvpc` Netzwerkmodus verwenden, von der Amazon ECS-Task.
- IAM — Sie können den Zugriff auf ein Amazon EFS-Dateisystem mithilfe von IAM einschränken. Wenn sie konfiguriert sind, benötigen Amazon ECS-Aufgaben eine IAM-Rolle für den

Dateisystemzugriff, um ein EFS-Dateisystem zu mounten. Weitere Informationen finden Sie unter Verwenden von IAM zur Steuerung des Dateisystemdatenzugriffs im Amazon Elastic File System-Benutzerhandbuch.

IAM-Richtlinien können auch vordefinierte Bedingungen erzwingen, z. B. die Anforderung, dass ein Client TLS verwendet, wenn er eine Verbindung zu einem Amazon EFS-Dateisystem herstellt. Weitere Informationen finden Sie unter Amazon EFS-Bedingungsschlüssel für Clients im Amazon Elastic File System-Benutzerhandbuch.

- Amazon EFS-Zugriffspunkte — Amazon EFS-Zugriffspunkte sind anwendungsspezifische Einstiegspunkte in ein Amazon EFS-Dateisystem. Sie können Access Points verwenden, um eine Benutzeridentität, einschließlich der POSIX-Gruppen des Benutzers, für alle Dateisystemanfragen, die über den Access Point gestellt werden, durchzusetzen. Zugriffspunkte können auch ein anderes Stammverzeichnis für das Dateisystem erzwingen. Auf diese Weise können Clients nur auf Daten im angegebenen Verzeichnis oder seinen Unterverzeichnissen zugreifen.

Erwägen Sie die Implementierung aller drei Zugriffskontrollen in einem Amazon EFS-Dateisystem, um maximale Sicherheit zu gewährleisten. Sie können beispielsweise die Sicherheitsgruppe, die einem Amazon EFS-Bereitstellungspunkt zugeordnet ist, so konfigurieren, dass nur eingehender NFS-Verkehr von einer Sicherheitsgruppe zugelassen wird, die Ihrer Container-Instance oder Amazon ECS-Aufgabe zugeordnet ist. Darüber hinaus können Sie Amazon EFS so konfigurieren, dass für den Zugriff auf das Dateisystem eine IAM-Rolle erforderlich ist, auch wenn die Verbindung aus einer zugelassenen Sicherheitsgruppe stammt. Schließlich können Sie Amazon EFS-Zugriffspunkte verwenden, um POSIX-Benutzerberechtigungen durchzusetzen und Stammverzeichnisse für Anwendungen anzugeben.

Der folgende Ausschnitt aus der Aufgabendefinition zeigt, wie ein Amazon EFS-Dateisystem mithilfe eines Access Points bereitgestellt wird.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "acessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
      },
```

```
        "name": "my-filesystem"
    }
]
```

Leistung des Amazon EFS-Volumes

Amazon EFS bietet zwei Leistungsmodi: General Purpose und Max I/O. General Purpose eignet sich für latenzempfindliche Anwendungen wie Content-Management-Systeme und CI/CD-Tools. Im Gegensatz dazu eignen sich Max I/O-Dateisysteme für Workloads wie Datenanalyse, Medienverarbeitung und maschinelles Lernen. Diese Workloads müssen parallel Operationen von Hunderten oder sogar Tausenden von Containern aus ausführen und erfordern den höchstmöglichen Gesamtdurchsatz und die höchstmöglichen IOPS. Weitere Informationen finden Sie unter Amazon EFS-Leistungsmodi im Amazon Elastic File System-Benutzerhandbuch.

Einige latenzempfindliche Workloads erfordern sowohl die höheren I/O-Stufen, die durch den Modus Max I/O Performance bereitgestellt werden, als auch die niedrigere Latenz, die durch den Allzweck-Performance-Modus bereitgestellt wird. Für diese Art von Workload empfehlen wir, mehrere Allzweck-Leistungsmodus-Dateisysteme zu erstellen. Auf diese Weise können Sie die Arbeitslast Ihrer Anwendung auf all diese Dateisysteme verteilen, sofern die Arbeitslast und die Anwendungen sie unterstützen können.

Durchsatz von Amazon EFS-Volumes

Allen Amazon EFS-Dateisystemen ist ein gemessener Durchsatz zugeordnet, der entweder durch die Menge des bereitgestellten Durchsatzes für Dateisysteme mit bereitgestelltem Durchsatz oder durch die Menge der in der EFS-Speicherklasse Standard oder One Zone gespeicherten Daten für Dateisysteme mit Bursting Throughput bestimmt wird. Weitere Informationen finden Sie unter Understanding Metered Throughput im Amazon Elastic File System-Benutzerhandbuch.

Der Standard-Durchsatzmodus für Amazon EFS-Dateisysteme ist der Bursting-Modus. Im Bursting-Modus wird der Durchsatz, der einem Dateisystem zur Verfügung steht, mit zunehmendem Dateisystem nach oben oder unten skaliert. Da dateibasierte Workloads in der Regel stark ansteigen und für bestimmte Zeiträume einen hohen Durchsatz und für den Rest der Zeit einen niedrigeren Durchsatz erfordern, ist Amazon EFS so konzipiert, dass es schnell geht, um hohe Durchsatzwerte für bestimmte Zeiträume zu ermöglichen. Da viele Workloads leseintensiv sind, werden Lesevorgänge außerdem im Verhältnis 1:3 zu anderen NFS-Vorgängen (wie Schreiben) gemessen.

Alle Amazon EFS-Dateisysteme bieten eine konsistente Basisleistung von 50 MB/s für jedes TB Amazon EFS Standard- oder Amazon EFS One Zone-Speicher. Alle Dateisysteme (unabhängig von

ihrer Größe) können bis zu 100 MB/s erreichen. Dateisysteme mit mehr als 1 TB EFS Standard- oder EFS One Zone-Speicher können für jedes TB auf 100 MB/s ansteigen. Da Lesevorgänge im Verhältnis 1:3 gemessen werden, können Sie bis zu 300 MiBs /s pro TiB Lesedurchsatz erreichen. Wenn Sie Ihrem Dateisystem Daten hinzufügen, wird der maximale Durchsatz, der für das Dateisystem verfügbar ist, linear und automatisch mit Ihrem Speicher in der Amazon EFS-Standardspeicherklasse skaliert. Wenn Sie mehr Durchsatz benötigen, als Sie mit Ihrer gespeicherten Datenmenge erreichen können, können Sie den bereitgestellten Durchsatz auf die spezifische Menge konfigurieren, die Ihre Arbeitslast benötigt.

Der Dateisystemdurchsatz wird von allen Amazon EC2 EC2-Instances gemeinsam genutzt, die mit einem Dateisystem verbunden sind. Beispielsweise kann ein 1-TB-Dateisystem, das einen Durchsatz von 100 MB/s erreichen kann, 100 MB/s aus einer einzigen Amazon EC2-Instance erzeugen, die jeweils 10 MB/s erreichen kann. Weitere Informationen finden Sie unter Amazon EFS-Leistung im Amazon Elastic File System-Benutzerhandbuch.

Optimierung der Kosten für Amazon EFS-Volumes

Amazon EFS vereinfacht die Speicherskalierung für Sie. Amazon EFS-Dateisysteme wachsen automatisch, wenn Sie mehr Daten hinzufügen. Insbesondere im Amazon EFS Bursting Throughput-Modus skaliert der Durchsatz auf Amazon EFS, wenn die Größe Ihres Dateisystems in der Standardspeicherklasse zunimmt. Um den Durchsatz zu verbessern, ohne zusätzliche Kosten für den bereitgestellten Durchsatz auf einem EFS-Dateisystem zu zahlen, können Sie ein Amazon EFS-Dateisystem mit mehreren Anwendungen gemeinsam nutzen. Mithilfe von Amazon EFS-Zugriffspunkten können Sie die Speicherisolierung in gemeinsam genutzten Amazon EFS-Dateisystemen implementieren. Auf diese Weise können die Anwendungen, obwohl sie immer noch dasselbe Dateisystem verwenden, nicht auf Daten zugreifen, es sei denn, Sie autorisieren sie.

Wenn Ihre Daten wachsen, hilft Ihnen Amazon EFS dabei, Dateien, auf die selten zugegriffen wird, automatisch in eine niedrigere Speicherklasse zu verschieben. Die Amazon EFS-Speicherklasse Standard-Infrequent Access (IA) reduziert die Speicherkosten für Dateien, auf die nicht täglich zugegriffen wird. Dies geschieht, ohne auf die hohe Verfügbarkeit, Haltbarkeit, Elastizität und den POSIX-Dateisystemzugriff zu verzichten, den Amazon EFS bietet. Weitere Informationen finden Sie unter Amazon EFS-Speicherklassen im Amazon Elastic File System-Benutzerhandbuch.

Erwägen Sie die Verwendung von Amazon EFS-Lebenszyklusrichtlinien, um automatisch Geld zu sparen, indem Sie selten aufgerufene Dateien in den Amazon EFS IA-Speicher verschieben. Weitere Informationen finden Sie unter Amazon-EFS-Lebenszyklusverwaltung im Benutzerhandbuch zu Amazon Elastic File System.

Bei der Erstellung eines Amazon EFS-Dateisystems können Sie wählen, ob Amazon EFS Ihre Daten über mehrere Availability Zones (Standard) hinweg repliziert oder Ihre Daten redundant in einer einzigen Availability Zone speichert. Die Amazon EFS One Zone-Speicherklasse kann die Speicherkosten im Vergleich zu Amazon EFS Standard-Speicherklassen erheblich senken. Erwägen Sie die Verwendung der Amazon EFS One Zone-Speicherklasse für Workloads, für die keine Multi-AZ-Resilienz erforderlich ist. Sie können die Kosten für Amazon EFS One Zone-Speicher weiter senken, indem Sie Dateien, auf die selten zugegriffen wird, nach Amazon EFS One Zone-Infrequent Access verschieben. Weitere Informationen finden Sie unter Amazon EFS Infrequent Access.

Amazon EFS Volumendatenschutz

Amazon EFS speichert Ihre Daten redundant in mehreren Availability Zones für Dateisysteme, die Standardspeicherklassen verwenden. Wenn Sie Amazon EFS One Zone-Speicherklassen auswählen, werden Ihre Daten redundant in einer einzigen Availability Zone gespeichert. Darüber hinaus ist Amazon EFS so konzipiert, dass es über ein bestimmtes Jahr eine Haltbarkeit von 99,999999999% (11 9) bietet.

Wie in jeder Umgebung ist es eine bewährte Methode, ein Backup zu erstellen und Schutzmaßnahmen gegen versehentliches Löschen zu treffen. Für Amazon EFS-Daten umfasst diese bewährte Methode ein funktionierendes, regelmäßig getestetes Backup mit AWS Backup. Dateisysteme, die Amazon EFS One Zone-Speicherklassen verwenden, sind so konfiguriert, dass Dateien bei der Erstellung des Dateisystems standardmäßig automatisch gesichert werden, sofern Sie diese Funktion nicht deaktivieren. Weitere Informationen finden Sie unter Datenschutz für Amazon EFS im Amazon Elastic File System-Benutzerhandbuch.

Geben Sie ein Amazon EFS-Dateisystem in einer Amazon ECS-Aufgabendefinition an

Um Amazon EFS-Dateisystem-Volumes für Ihre Container zu verwenden, müssen Sie die Volume- und Bereitstellungspunkt-Konfigurationen in der Aufgabendefinition angeben. Das folgende JSON-Codefragment der Aufgabendefinition zeigt die Syntax für die Objekte `volumes` und `mountPoints` für einen Container.

```
{
    "containerDefinitions": [
        {
            "name": "container-using-efs",
            "image": "amazonlinux:2",
            "entryPoint": [
                "sh",
                "-c"
```

```
            ],
            "command": [
                "ls -la /mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEfsVolume",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1234",
                "rootDirectory": "/path/to/my/data",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": integer,
                "authorizationConfig": {
                    "accessPointId": "fsap-1234",
                    "iam": "ENABLED"
                }
            }
        }
    ]
}
```

`efsVolumeConfiguration`

Typ: Objekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von Amazon EFS-Volumes angegeben.

`fileSystemId`

Typ: Zeichenfolge

Erforderlich: Ja

Die zu verwendende Amazon EFS-Dateisystem-ID.

`rootDirectory`

Typ: Zeichenfolge

Erforderlich: Nein

Das Verzeichnis im Amazon EFS-Dateisystem, das als Stammverzeichnis im Host bereitgestellt werden soll. Wenn dieser Parameter weggelassen wird, wird der Stamm des Amazon-EFS-Volumes verwendet. Die Angabe von `/` hat denselben Effekt wie das Weglassen dieses Parameters.

> **Important**
>
> Wenn ein EFS-Zugriffspunkt in `authorizationConfig` angegeben ist, muss der Stammverzeichnis-Parameter entweder weggelassen oder auf `/` festgelegt werden, was erzwingt, dass der Pfad für den EFS-Zugriffspunkt festgelegt wird.

`transitEncryption`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Gibt an, ob die Verschlüsselung für Amazon-EFS-Daten während der Übertragung zwischen dem Amazon-ECS-Host und dem Amazon-EFS-Server aktiviert werden soll oder nicht. Wenn die Amazon-EFS-IAM-Autorisierung verwendet wird, muss die Transit-Verschlüsselung aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter Verschlüsseln von Daten während der Übertragung im Benutzerhandbuch für Amazon Elastic File System.

`transitEncryptionPort`

Typ: Ganzzahl

Erforderlich: Nein

Der zu verwendende Port zum Senden verschlüsselter Daten zwischen dem Amazon-ECS-Host und dem Amazon EFS-Server. Wenn Sie keinen Transit-Verschlüsselungsport angeben,

wird die Port-Auswahlstrategie verwendet, die der Amazon EFS-Mount-Helfer verwendet. Weitere Informationen finden Sie unter EFS-Mount-Helfer im Benutzerhandbuch für Amazon Elastic File System.

`authorizationConfig`

Typ: Objekt

Erforderlich: Nein

Die Autorisierungskonfigurationsdetails für das Amazon EFS-Dateisystem.

`accessPointId`

Typ: Zeichenfolge

Erforderlich: Nein

Die zu verwendende Zugriffspunkt-ID. Wenn ein Zugriffspunkt angegeben wird, muss der Stammverzeichniswert in `efsVolumeConfiguration` ausgelassen oder auf `/` festgelegt werden, was den auf dem EFS-Zugriffspunkt festgelegten Pfad erzwingt. Wenn ein Zugriffspunkt verwendet wird, muss in `EFSVolumeConfiguration` die Transitverschlüsselung aktiviert sein. Weitere Informationen finden Sie unter Arbeiten mit Amazon EFS-Zugriffspunkten im Amazon Elastic File System-Benutzerhandbuch.

`iam`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Gibt an, ob die in einer Aufgabendefinition definierte Amazon-ECS-Aufgaben-IAM-Rolle beim Mounten des Amazon EFS-Dateisystems verwendet werden soll. Wenn diese Option aktiviert ist, muss die Transit-Verschlüsselung in `EFSVolumeConfiguration` aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben.

Konfiguration von Amazon EFS-Dateisystemen für Amazon ECS mithilfe der Konsole

Erfahren Sie, wie Sie Amazon Elastic File System (Amazon EFS) -Dateisysteme mit Amazon ECS verwenden.

Schritt 1: Erstellen eines Amazon-ECS-Clusters

Führen Sie die folgenden Schritte aus, um einen Amazon-ECS-Cluster zu erstellen.

So erstellen Sie einen neuen Cluster (Amazon ECS-Konsole)

Bevor Sie beginnen, weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Geben Sie unter Cluster-Konfiguration für Cluster-Name `EFS-tutorial` für den Cluster-Namen ein.
6. (Optional) Um die VPC und Subnetze zu ändern, in denen Ihre Aufgaben und Services gelauncht werden, führen Sie unter Networking (Netzwerk) einen der folgenden Vorgänge aus:
   - Um ein Subnetz zu entfernen, wählen Sie unter Subnets (Subnetze) X für jedes Subnetz, das Sie entfernen möchten.
   - Um zu einer anderen VPC als der Standard-VPC zu wechseln, wählen Sie unter VPC eine vorhandene VPC und dann unter Subnets (Subnetze) jedes Subnetz aus.
7. Um Ihrem Cluster Amazon-EC2-Instances hinzuzufügen, erweitern Sie Infrastruktur und wählen Sie dann Amazon-EC2-Instances aus. Konfigurieren Sie als Nächstes die Auto-Scaling-Gruppe, die als Kapazitätsanbieter fungiert:
   - Um eine Auto-Scaling-Gruppe zu erstellen, wählen Sie in der Auto-Scaling-Gruppe (ASG) Create new group (Neue Gruppe erstellen) und geben Sie dann die folgenden Details zur Gruppe an:
     - Wählen Sie für Betriebssystem/Architektur die Option Amazon Linux 2 aus.
     - Wählen Sie unter EC2 instance type (EC2-Instance-Typ) die Option `t2.micro` aus.

       Wählen Sie für SSH key pair (SSH-Schlüsselpaar) das Paar aus, das Ihre Identität nachweist, wenn Sie eine Verbindung zur Instance herstellen.
     - Geben Sie für Kapazität den Wert 1 ein.
8. Wählen Sie Erstellen.

Schritt 2: Eine Sicherheitsgruppe für Amazon-EC2-Instances und das Amazon-EFS-Dateisystem erstellen

In diesem Schritt erstellen Sie eine Sicherheitsgruppe für Ihre Amazon-EC2-Instances, die eingehenden Netzwerkverkehr auf Port 80 zulässt, und Ihr Amazon-EFS-Dateisystem, das den eingehenden Zugriff von Ihren Container-Instances erlaubt.

Erstellen Sie eine Sicherheitsgruppe für Ihre Amazon-EC2-Instances mit den folgenden Optionen:

- Sicherheitsgruppenname – Ein eindeutiger Name für Ihre Sicherheitsgruppe.
- VPC – Die VPC, die Sie zuvor für Ihr Cluster festgelegt haben.
- Regel für eingehenden Datenverkehr
  - Typ – HTTP
  - Quelle – 0.0.0.0/0.

Erstellen Sie eine Sicherheitsgruppe für Ihr Amazon-EFS-Dateisystem mit den folgenden Optionen:

- Sicherheitsgruppenname – Ein eindeutiger Name für Ihre Sicherheitsgruppe. z. B. `EFS-access-for-sg-`*`dc025fa2`*.
- VPC – Die VPC, die Sie zuvor für Ihr Cluster festgelegt haben.
- Regel für eingehenden Datenverkehr
  - Typ – NFS
  - Quelle – Benutzerdefiniert mit der ID der Sicherheitsgruppe, die Sie für Ihre Instances erstellt haben.

Informationen zum Erstellen einer Sicherheitsgruppe finden Sie unter Erstellen einer Sicherheitsgruppe im Amazon EC2 EC2-Benutzerhandbuch.

Schritt 3: Erstellen eines Amazon EFS-Dateisystems

In diesem Schritt erstellen Sie ein Amazon EFS-Dateisystem.

Erstellen eines Amazon EFS Dateisystems für Amazon-ECS-Aufgaben.

1. Öffnen Sie die Amazon Elastic File System-Konsole unter https://console.aws.amazon.com/efs/.
2. Wählen Sie Create file system (Dateisystem erstellen) aus.

3. Geben Sie einen Namen für Ihr Dateisystem ein und wählen Sie dann die VPC aus, in der Ihre Container-Instances gehostet werden. Standmäßig erhält jedes Subnetz in der angegebenen VPC ein Mountingziel, das die Standardsicherheitsgruppe für diese VPC verwendet. Wählen Sie dann Anpassen aus.

   > **Note**
   >
   > In diesem Tutorial wird davon ausgegangen, dass sich Ihr Amazon EFS-Dateisystem, Ihr Amazon ECS-Cluster, Ihre Container-Instances und Ihre Aufgaben in derselben VPC befinden. Weitere Informationen zum Mounten eines Dateisystems von einer anderen VPC finden Sie unter Exemplarische Vorgehensweise: Mounten eines Dateisystems aus einer anderen VPC im Amazon EFS-Benutzerhandbuch.

4. Konfigurieren Sie auf der Seite mit den Dateisystemeinstellungen optionale Einstellungen und wählen Sie dann unter Leistungseinstellungen den Bursting-Durchsatzmodus für Ihr Dateisystem aus. Nachdem Sie die Einstellungen konfiguriert haben, wählen Sie Weiter aus.

   a. (Optional) Fügen Sie Tags für Ihr Dateisystem hinzu. Sie können beispielweise einen eindeutigen Namen für das Dateisystem angeben, indem Sie den entsprechenden Namen in der Spalte Value neben dem Schlüssel Name eingeben.

   b. (Optional) Aktivieren Sie die Lebenszyklusverwaltung, um Kosten bei selten genutztem Speicher zu sparen. Weitere Informationen finden Sie unter EFS lifecycle management (EFS-Lebenszyklusverwaltung) im Amazon Elastic File System-Benutzerhandbuch.

   c. (Optional) Aktivieren Sie die Verschlüsselung. Wählen Sie das Kontrollkästchen aus, um die Verschlüsselung Ihres Amazon EFS-Dateisystems im Ruhezustand zu aktivieren.

5. Ersetzen Sie auf der Seite Netzwerkzugriff unter Mount-Ziele die bestehende Sicherheitsgruppenkonfiguration für jede Availability Zone durch die Sicherheitsgruppe, die Sie für das Dateisystem in Schritt 2: Eine Sicherheitsgruppe für Amazon-EC2-Instances und das Amazon-EFS-Dateisystem erstellen erstellt haben, und wählen Sie dann Weiter aus.

6. Sie müssen für dieses Tutorial keine Dateisystemrichtlinie konfigurieren, sodass Sie den Abschnitt überspringen können, indem Sie Weiter wählen.

7. Überprüfen Sie die Dateisystemoptionen und wählen Sie Erstellen, um den Vorgang abzuschließen.

8. Notieren Sie sich auf dem Bildschirm Dateisysteme die Dateisystem-ID. Im nächsten Schritt referenzieren Sie diesen Wert in der Amazon-ECS-Aufgabendefinition.

Schritt 4: Hinzufügen von Inhalten zum Amazon EFS-Dateisystem

In diesem Schritt mounten Sie das Amazon EFS-Dateisystem einer Amazon-EC2-Instance und fügen Inhalte hinzu. Dies ist für Testzwecke in diesem Tutorial gedacht, um die persistente Natur der Daten zu veranschaulichen. Wenn Sie dieses Feature verwenden, verwenden Sie normalerweise Ihre Anwendung oder eine andere Methode, mit der Sie Daten in Ihr Amazon-EFS-Dateisystem schreiben.

So erstellen Sie eine Amazon-EC2-Instance und stellen das Amazon EFS-Dateisystem bereit

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie Launch Instance aus.
3. Wählen Sie unter Anwendungs- und Betriebssystem-Images (Amazon Machine Image) das Amazon Linux 2 AMI (HVM).
4. Behalten Sie unter Instance-Typ den Standard-Instance-Typ, `t2.micro` bei.
5. Wählen Sie unter Schlüsselpaar (Anmeldung) ein Schlüsselpaar für den SSH-Zugriff auf die Instance aus.
6. Wählen Sie unter Netzwerkeinstellungen, die VPC aus, die Sie für Ihr Amazon-EFS-Dateisystem und Ihren Amazon-ECS-Cluster angegeben haben. Wählen Sie ein Subnetz und die Instance-Sicherheitsgruppe aus, die in Schritt 2: Eine Sicherheitsgruppe für Amazon-EC2-Instances und das Amazon-EFS-Dateisystem erstellen erstellt wurde. Konfigurieren Sie die Sicherheitsgruppe der Instance. Vergewissern Sie sich, dass Öffentliche IP automatisch zuweisen aktiviert ist.
7. Wählen Sie unter Speicher konfigurieren die Schaltfläche Bearbeiten für Dateisysteme und wählen Sie dann EFS. Wählen Sie das Dateisystem aus, das Sie in Schritt 3: Erstellen eines Amazon EFS-Dateisystems erstellt haben. Sie können optional den Mounting-Punkt ändern oder den Standardwert belassen.

   > **Important**
   >
   > Sie müssen ein Subnetz auswählen, bevor Sie ein Dateisystem zur Instance hinzufügen können.

8. Deaktivieren Sie die Option Sicherheitsgruppen automatisch erstellen und anhängen. Lassen Sie das andere Kontrollkästchen aktiviert. Wählen Sie Add shared file system (Freigegebenes Dateisystem hinzufügen) aus.
9. Stellen Sie unter Advanced Details (Erweiterte Details) sicher, dass das Benutzerdatenskript automatisch mit den Mounting-Schritten des Amazon EFS-Dateisystems gefüllt wird.

10. Stellen Sie unter Zusammenfassung sicher, dass die Anzahl der Instances 1 beträgt. Wählen Sie Launch Instance (Instance starten) aus.
11. Wählen Sie auf der Seite Eine Instance starten die Option Alle Instances anzeigen, um den Status Ihrer Instances zu sehen. Anfänglich ist der Instance-Zustand `PENDING`. Nachdem sich der Status auf `RUNNING` geändert hat und die Instance alle Statusprüfungen bestanden hat, ist die Instance einsatzbereit.

Stellen Sie jetzt eine Verbindung zur Amazon-EC2-Instance her und fügen dem Amazon EFS Dateisystem Inhalt hinzu.

So stellen Sie eine Verbindung zur Amazon-EC2-Instance her und fügen dem Amazon EFS Dateisystem Inhalt hinzu

1. Sie SSH auf die von Ihnen erstellte Amazon-EC2-Instance. Weitere Informationen finden Sie unter Connect to Your Linux Instance im Amazon EC2 EC2-Benutzerhandbuch.
2. Führen Sie im Terminalfenster den Befehl df -T aus, um zu überprüfen, ob das Amazon-EFS-Dateisystem gemountet wurde. In der folgenden Ausgabe haben wir das Mounting des Amazon EFS-Dateisystems hervorgehoben.

```
$ df -T
Filesystem     Type            1K-blocks    Used        Available Use% Mounted on
devtmpfs       devtmpfs           485468       0           485468   0% /dev
tmpfs          tmpfs              503480       0           503480   0% /dev/shm
tmpfs          tmpfs              503480     424           503056   1% /run
tmpfs          tmpfs              503480       0           503480   0% /sys/fs/
cgroup
/dev/xvda1     xfs               8376300 1310952          7065348  16% /
127.0.0.1:/    nfs4     9007199254739968       0 9007199254739968   0% /mnt/efs/fs1
tmpfs          tmpfs              100700       0           100700   0% /run/
user/1000
```

3. Navigieren Sie zu dem Verzeichnis, in dem das Amazon EFS-Dateisystem eingehängt ist. Im obigen Beispiel ist es `/mnt/efs/fs1`.
4. Erstellen Sie eine Datei mit dem Namen `index.html` und dem folgenden Inhalt:

```
<html>
    <body>
        <h1>It Works!</h1>
```

```
        <p>You are using an Amazon EFS file system for persistent container
 storage.</p>
    </body>
</html>
```

Schritt 5: Erstellen einer Aufgabendefinition

Die folgende Aufgabendefinition erstellt ein Daten-Volume mit dem Namen `efs-html`. Der `nginx`-Container mountet das Host-Daten-Volume im NGINX-Stammverzeichnis, `/usr/share/nginx/html`.

So erstellen Sie eine neue Aufgabendefinition mithilfe der Amazon-ECS-Konsole

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
4. Kopieren Sie im Feld JSON-Editor den folgenden JSON-Text und fügen Sie ihn ein, wobei Sie das `fileSystemId` durch die ID Ihres Amazon-EFS-Dateisystems ersetzen.

```
{
    "containerDefinitions": [
        {
            "memory": 128,
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "mountPoints": [
                {
                    "containerPath": "/usr/share/nginx/html",
                    "sourceVolume": "efs-html"
                }
            ],
            "name": "nginx",
            "image": "nginx"
```

```
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws::iam::111122223333:role/ecsTaskExecutionRole"
}
```

> **Note**
>
> Sie können Ihrer IAM-Rolle für die Ausführung von Amazon ECS-Aufgaben die folgenden Berechtigungen hinzufügen, damit der Amazon ECS-Agent beim Start ein Amazon EFS-Dateisystem finden und für eine Aufgabe bereitstellen kann.
>
> - `elasticfilesystem:ClientMount`
> - `elasticfilesystem:ClientWrite`
> - `elasticfilesystem:DescribeMountTargets`
> - `elasticfilesystem:DescribeFileSystems`

5. Wählen Sie Erstellen.

Schritt 6: Eine Aufgabe ausführen und die Ergebnisse anzeigen

Nachdem Ihr Amazon EFS-Dateisystem erstellt wurde und Webinhalte für den NGINX-Container bereitgestellt werden können, können Sie eine Aufgabe mithilfe der von Ihnen erstellten Aufgabendefinition ausführen. Der NGINX-Webserver stellt eine einfache HTML-Seite bereit. Wenn Sie den Inhalt in Ihrem Amazon EFS-Dateisystem aktualisieren, werden diese Änderungen an alle weiteren Container weitergegeben, auf denen dieses Dateisystem ebenfalls gemountet ist.

Die Aufgabe wird in dem Subnetz ausgeführt, das Sie für den Cluster definiert haben.

So führen Sie eine Aufgabe aus und zeigen die Ergebnisse mit der Konsole an

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Seite Clusters den Cluster aus, der die eigenständige Aufgabe ausführen soll.

   Bestimmen Sie die Ressource, von der aus Sie den Service starten.

So starten Sie einen Service von	Schritte
Cluster	a. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie im Service erstellen möchten. b. Von der Registerkarte Tasks (Aufgaben) wählen Sie Ausführen einer neuen Aufgabe.
Starttyp	a. Wählen Sie auf der Seite Task (Aufgabe) die Aufgabendefinition aus. b. Wenn mehr als eine Revision vorhanden ist, wählen Sie die Revision aus. c. Wählen Create (Erstelle n), Run task (Aufgabe ausführen).

3. (Optional) Wählen Sie aus, wie Ihre geplante Aufgabe auf Ihre Cluster-Infrastruktur aufgeteilt ist. Erweitern Sie die Compute configuration (Datenverarbeitungskonfiguration) und tun Sie folgendes:

Verteilungsmethode	Schritte
Starttyp	a. Wählen Sie im Bereich Compute options (Datenverarbeitungs-Optionen) die Option Launch type (Starttyp) aus. b. Wählen Sie unter Starttyp EC2 aus.

4. Für Anwendungstyp, wählen Sie Aufgabe aus.
5. Wählen Sie für Aufgabendefinition die `efs-tutorial`-Aufgabendefinition aus, die Sie zuvor erstellt haben.
6. Geben Sie für Gewünschte Aufgaben 1 ein.
7. Wählen Sie Erstellen.
8. Wählen Sie auf der Seite Cluster die Registerkarte Infrastruktur.
9. Wählen Sie unter Container-Instances die Container-Instance aus, zu der Sie eine Verbindung herstellen möchten.
10. Notieren Sie auf der Seite Container-Instance unter Netzwerk die öffentliche IP-Adresse für Ihre Instance.
11. Öffnen Sie einen Browser und geben Sie die öffentliche IP-Adresse ein. Sie sollten folgende Meldung sehen:

```
It works!
You are using an Amazon EFS file system for persistent container storage.
```

> **Note**
>
> Wenn Sie die Meldung nicht sehen, vergewissern Sie sich, dass die Sicherheitsgruppe für Ihre Container-Instance den eingehenden Netzwerkverkehr auf Port 80 und die Sicherheitsgruppe für Ihr Dateisystem den eingehenden Zugriff von der Container-Instance erlaubt.

Verwenden Sie FSx for Windows File Server Windows-Dateiserver-Volumes mit Amazon ECS

FSx for Windows File Server bietet vollständig verwaltete Windows-Dateiserver, die von einem Windows-Dateisystem unterstützt werden. Wenn Sie FSx for Windows File Server zusammen mit ECS verwenden, können Sie Ihre Windows-Aufgaben mit persistenter, verteilter, freigegebener, statischer Dateispeicher bereitstellen. Weitere Informationen finden Sie unter Was ist FSx for Windows File Server?.

> **Note**
>
> EC2-Instances, die das Amazon-ECS-optimierte Windows Server 2016 Full AMI verwenden, unterstützen FSx for Windows File Server ECS-Aufgaben-Volumes nicht.
> Sie können FSx nicht für Windows-Dateiserver-Volumes in einer Windows-Container-On-Fargate-Konfiguration verwenden. Stattdessen können Sie Container so ändern, dass sie beim Start bereitgestellt werden.

Sie können FSx for Windows File Server verwenden, um Windows-Arbeitslasten bereitzustellen, die Zugriff auf gemeinsam genutzten externen Speicher, hochverfügbaren regionalen Speicher oder Massenspeicher mit hohem Durchsatz erfordern. Sie können ein oder mehrere Dateisystem-Volumes vom Typ FSx for Windows File Server in einen Amazon ECS-Container einbinden, der auf einer Amazon ECS-Windows-Instance ausgeführt wird. Sie können Dateisystem-Volumes von FSx for Windows File Server zwischen mehreren Amazon ECS-Containern innerhalb einer einzigen Amazon ECS-Aufgabe gemeinsam nutzen.

Um die Verwendung von FSx for Windows File Server mit ECS zu aktivieren, müssen Sie die FSx for Windows File Server-Dateisystem-ID und zugehörige Informationen in eine Aufgabendefinition einschließen. Das wird im folgenden JSON-Snippet-Beispiel für Aufgabendefinition veranschaulicht. Bevor Sie eine Aufgabendefinition erstellen und ausführen, benötigen Sie Folgendes.

- Eine ECS-Windows EC2-Instance, die mit einer gültigen Domain verbunden ist. Es kann von einem lokalen Active Directory oder einem AWS Directory Service for Microsoft Active Directoryselbst gehosteten Active Directory auf Amazon EC2 gehostet werden.
- Ein AWS Secrets Manager geheimer oder Systems Manager Manager-Parameter, der die Anmeldeinformationen enthält, die verwendet werden, um der Active Directory-Domäne beizutreten und das Dateisystem FSx for Windows File Server anzuhängen. Bei den

Anmeldeinformationswerten handelt es sich um den Namen und das Kennwort, die Sie beim Erstellen von Active Directory eingegeben haben.

Ein dazugehöriges Tutorial finden Sie unter Erfahren Sie, wie Sie FSx for Windows File Server Server-Dateisysteme für Amazon ECS konfigurieren.

Überlegungen

Beachten Sie Folgendes, wenn Sie FSx for Windows File Server Volumes verwenden:

- FSx for Windows File Server mit Amazon ECS unterstützt nur Windows Amazon-EC2-Instances. Amazon-EC2-Instances werden nicht unterstützt.
- FSx for Windows File Server mit Amazon ECS unterstützt AWS Fargate nicht.
- FSx for Windows File Server mit Amazon ECS mit `awsvpc`-Netzwerkmodus erfordert Version `1.54.0` oder höher des Container-Agenten.
- Die maximale Anzahl von Laufwerksbuchstaben, die für eine Amazon-ECS-Aufgabe verwendet werden können, beträgt 23. Jeder Task mit einem FSx for Windows File Server Volume erhält einen Laufwerkbuchstaben zugewiesen.
- Die Bereinigungszeit der Aufgabenressource beträgt standardmäßig 3 Stunden nach Beendigung der Aufgabe. Eine Dateizuordnung, die von einer Aufgabe erstellt wurde, bleibt für 3 Stunden bestehen, selbst wenn keine Aufgaben sie verwenden. Die standardmäßige Bereinigungszeit kann mithilfe der Amazon-ECS-Umgebungsvariablen `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` konfiguriert werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.
- In der Regel werden Aufgaben nur in derselben VPC ausgeführt wie das FSx for Windows File Server-Dateisystem. Es ist jedoch möglich, Cross-VPC-Unterstützung zu haben, wenn eine etablierte Netzwerkverbindung zwischen der Amazon-ECS-Cluster-VPC und dem FSx for Windows File Server-Dateisystem über VPC-Peering besteht.
- Sie steuern den Zugriff auf ein FSx for Windows File Server-Dateisystem auf Netzwerkebene, indem Sie die VPC-Sicherheitsgruppen konfigurieren. Nur Aufgaben, die auf EC2-Instances gehostet werden, die zur Active Directory-Domäne mit korrekt konfigurierten Active Directory-Sicherheitsgruppen gehören, können auf die FSx for Windows File Server Server-Dateifreigabe zugreifen. Wenn die Sicherheitsgruppen falsch konfiguriert sind, kann Amazon ECS die Aufgabe nicht mit der folgenden Fehlermeldung starten:`unable to mount file system` *`fs-id`*.“
- FSx for Windows File Server ist in AWS Identity and Access Management (IAM) integriert, um die Aktionen zu steuern, die Ihre IAM-Benutzer und -Gruppen mit bestimmten FSx for Windows

File Server Server-Ressourcen ausführen können. Mit der Clientautorisierung können Kunden IAM-Rollen definieren, die den Zugriff auf bestimmte FSx for Windows File Server-Dateisysteme zulassen oder verweigern, optional einen schreibgeschützten Zugriff erfordern und optional den Root-Zugriff auf das Dateisystem vom Client aus zulassen oder verbieten. Weitere Informationen finden Sie in Sicherheit im Amazon FSx for Windows File Server-Benutzerhandbuch.

Bewährte Methoden für die Verwendung von FSx for Windows File Server mit Amazon ECS

Beachten Sie die folgenden Best-Practice-Empfehlungen, wenn Sie FSx for Windows File Server mit Amazon ECS verwenden.

Sicherheits- und Zugriffskontrollen für FSx for Windows File Server

FSx for Windows File Server bietet die folgenden Funktionen zur Zugriffskontrolle, mit denen Sie sicherstellen können, dass die in einem FSx for Windows File Server Server-Dateisystem gespeicherten Daten sicher sind und nur von Anwendungen aus zugänglich sind, die sie benötigen.

Datenverschlüsselung für FSx for Windows File Server Server-Volumes

FSx for Windows File Server unterstützt zwei Formen der Verschlüsselung für Dateisysteme. Dabei handelt es sich um die Verschlüsselung von Daten während der Übertragung und die Verschlüsselung im Ruhezustand. Die Verschlüsselung von Daten während der Übertragung wird auf Dateifreigaben unterstützt, die einer Container-Instance zugeordnet sind, die das SMB-Protokoll 3.0 oder höher unterstützt. Die Verschlüsselung von Daten im Ruhezustand wird automatisch aktiviert, wenn ein Amazon FSx-Dateisystem erstellt wird. Amazon FSx verschlüsselt Daten während der Übertragung automatisch mit SMB-Verschlüsselung, wenn Sie auf Ihr Dateisystem zugreifen, ohne dass Sie Ihre Anwendungen ändern müssen. Weitere Informationen finden Sie unter Datenverschlüsselung in Amazon FSx im Amazon FSx for Windows File Server Server-Benutzerhandbuch.

Verwenden Sie Windows-ACLs für die Zugriffskontrolle auf Ordnerebene

Die Windows Amazon EC2 EC2-Instance greift mithilfe von Active Directory-Anmeldeinformationen auf Amazon FSx-Dateifreigaben zu. Sie verwendet standardmäßige Windows-Zugriffskontrolllisten (ACLs) für eine differenzierte Zugriffskontrolle auf Datei- und Ordnerebene. Sie können mehrere Anmeldeinformationen erstellen, jeweils für einen bestimmten Ordner innerhalb der Freigabe, der einer bestimmten Aufgabe zugeordnet ist.

Im folgenden Beispiel hat die Aufgabe `App01` mithilfe der in Secrets Manager gespeicherten Anmeldeinformationen Zugriff auf den Ordner. Sein Amazon-Ressourcenname (ARN) lautet`1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

In einem anderen Beispiel hat eine Aufgabe Zugriff auf den Ordner App02 mithilfe von Anmeldeinformationen, die im Secrets Manager gespeichert sind. Ihr ARN lautet 6789.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Geben Sie ein Dateisystem FSx for Windows File Server in einer Amazon ECS-Aufgabendefinition an

Um FSx for Windows File Server-Dateisystem-Volumes für Ihre Container zu verwenden, geben Sie die Volume- und Bereitstellungspunkt-Konfigurationen in der Aufgabendefinition an. Das folgende JSON-Codefragment der Aufgabendefinition zeigt die Syntax für die Objekte `volumes` und `mountPoints` für einen Container.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>It Works!</h2> <p>You are using Amazon
 FSx for Windows File Server file system for persistent container storage.</p>' -
Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
```

```
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force; Start-
Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -Destination C:\
\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
```

```
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

`FSxWindowsFileServerVolumeConfiguration`

Typ: Objekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von FSx for Windows File Server-Dateisystem für die Aufgabenspeicherung angegeben.

`fileSystemId`

Typ: Zeichenfolge

Erforderlich: Ja

Die zu verwendende FSx for Windows File Server Dateisystem-ID.

`rootDirectory`

Typ: Zeichenfolge

Erforderlich: Ja

Das Verzeichnis im FSx for Windows File Server-Dateisystem, das als Stammverzeichnis im Host aufgespielt werden soll.

`authorizationConfig`

`credentialsParameter`

Typ: Zeichenfolge

Erforderlich: Ja

Die Optionen für Autorisierungsanmeldeinformationen:

- Amazon-Ressourcenname (ARN) eines Secrets Manager-Secrets.
- Amazon-Ressourcenname (ARN) eines Systems Manager-Parameters.

`domain`

Typ: Zeichenfolge

Erforderlich: Ja

Ein vollqualifizierter Domänenname, der in einem AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) -Verzeichnis oder einem selbst gehosteten EC2-Active Directory gehostet wird.

Methoden zum Speichern von FSx for Windows File Server Server-Volumenanmeldedaten

Es gibt zwei verschiedene Methoden zum Speichern von Anmeldeinformationen für die Verwendung mit dem Anmeldeinformationen-Parameter.

- AWS Secrets Manager geheim

  Diese Anmeldeinformationen können in der AWS Secrets Manager Konsole mithilfe der Kategorie Andere geheime Daten erstellt werden. Sie fügen für jedes Schlüssel-/Wert-Paar, Benutzername/admin und Passwort/*password* eine Zeile hinzu.

- Systems Manager-Parameter

  Diese Anmeldeinformationen können in der Systems Manager-Parameterkonsole erstellt werden, indem Sie Text in das Formular eingeben, das im folgenden Codeausschnitt gezeigt wird.

```
{
  "username": "admin",
  "password": "password"
}
```

`credentialsParameter` im Aufgabendefinitions-Parameter `FSxWindowsFileServerVolumeConfiguration` enthält entweder den geheimen ARN oder den Systems Manager Parameter ARN. Weitere Informationen finden Sie unter Was ist AWS -Secrets Manager im Secrets Manager-Benutzerhandbuch und Systems Manager Parameter Store aus dem Systems Manager-Benutzerhandbuch.

Erfahren Sie, wie Sie FSx for Windows File Server Server-Dateisysteme für Amazon ECS konfigurieren

Erfahren Sie, wie Sie eine Amazon ECS-optimierte Windows-Instance starten, die ein Dateisystem FSx for Windows File Server und Container hostet, die auf das Dateisystem zugreifen können. Dazu erstellen Sie zunächst ein AWS Directory Service AWS verwaltetes Microsoft Active Directory. Anschließend erstellen Sie ein Dateisystem und einen Cluster mit FSx for Windows File Server File Server mit einer Amazon EC2 EC2-Instance und einer Aufgabendefinition. Sie konfigurieren die Aufgabendefinition für Ihre Container, damit Sie das FSx for Windows File Server-Dateisystem verwenden können. Zum Schluss testen Sie das Dateisystem.

Jedes Mal, wenn Sie das Active Directory oder das FSx for Windows File Server-Dateisystem starten oder löschen, dauert 20 bis 45 Minuten. Seien Sie bereit, mindestens 90 Minuten zu reservieren, um das Tutorial abzuschließen oder das Tutorial über ein paar Sitzungen abzuschließen.

Voraussetzungen für das Tutorial

- Ein Administratorbenutzer. Siehe Einrichtung für die Verwendung von Amazon ECS.
- (Optional) Ein `PEM`-Schlüsselpaar für die Verbindung mit Ihrer EC2-Windows-Instance über RDP-Zugriff. Weitere Informationen zum Erstellen eines Schlüsselpaars finden Sie unter Amazon EC2-Schlüsselpaare und Windows-Instances im Benutzerhandbuch für Windows-Instances.
- Eine VPC mit mindestens einem öffentlichen und einem privaten Subnetz sowie einer Sicherheitsgruppe. Sie können Ihre Standard-VPC verwenden. Sie benötigen kein NAT-Gateway oder -Gerät. AWS Directory Service unterstützt nicht Network Address Translation (NAT) mit Active Directory. Damit dies funktioniert, müssen sich das Active Directory, das Dateisystem für FSx für Windows File Server, der ECS-Cluster und die EC2-Instance in Ihrer VPC befinden. Weitere Informationen zu VPCs und Active Directories finden Sie unter Konfigurationen des Amazon VPC-Konsolenassistenten und Voraussetzungen für AWS -Managed Microsoft AD.
- Die IAM ecsInstanceRole - und ecsTaskExecution Rollenberechtigungen sind mit Ihrem Konto verknüpft. Diese serviceverknüpften Rollen ermöglichen es Services, API-Aufrufe durchzuführen und in Ihrem Namen auf Container, Secrets, Verzeichnisse und Dateiserver zuzugreifen.

Schritt 1: Erstellen von IAM-Zugriffsrollen

Erstellen Sie einen Cluster mit AWS Management Console.

1. Prüfen IAM-Rolle für Amazon-ECS-Container-Instance Sie, ob Sie über ein Konto verfügen, ecsInstanceRole und finden Sie heraus, wie Sie eines erstellen können, falls Sie noch keines haben.
2. Es wird empfohlen, Rollenrichtlinien für Mindestberechtigungen in einer tatsächlichen Produktionsumgebung anzupassen. Stellen Sie zum Durcharbeiten dieses Tutorials sicher, dass die folgende AWS verwaltete Richtlinie an Ihr ECS angehängt istInstanceRole. Fügen Sie die Richtlinie hinzu, wenn sie noch nicht angehängt ist.

   - Amazon EC2 EC2-Rolle ContainerServicefor
   - Amazon ManagedInstance SSM-Kern
   - AmazonSSM-Zugriff DirectoryService

   Um AWS verwaltete Richtlinien anzuhängen.

   a. Öffnen Sie die IAM-Konsole.
   b. Wählen Sie im Navigationsbereich Roles aus.
   c. Wählen Sie eine AWS -verwaltete Rolle.
   d. Wählen Sie Berechtigungen, Richtlinien anfügen.
   e. Um die verfügbaren Richtlinien zum Anfügen einzugrenzen, verwenden Sie Filter.
   f. Wählen Sie die entsprechende Richtlinie aus und wählen Sie dann Attach Policy (Richtlinie anfügen).
3. Hier IAM-Rolle für die Amazon-ECS-Aufgabenausführung erfahren Sie, ob Sie über ein ECS verfügen, TaskExecutionRole und wie Sie eines erstellen können, falls Sie noch keines haben.

   Es wird empfohlen, Rollenrichtlinien für Mindestberechtigungen in einer tatsächlichen Produktionsumgebung anzupassen. Stellen Sie beim Durcharbeiten dieses Tutorials sicher, dass die folgenden AWS verwalteten Richtlinien mit Ihrer TaskExecution ecs-Rolle verknüpft sind. Fügen Sie die Richtlinien hinzu, wenn sie noch nicht angehängt sind. Verwenden Sie das im vorherigen Abschnitt beschriebene Verfahren, um die AWS verwalteten Richtlinien anzuhängen.

   - SecretsManagerReadWrite
   - Amazon F SxRead OnlyAccess

- AmazonSSM-Zugriff ReadOnly
- AmazonECS TaskExecution RolePolicy

Schritt 2: Erstellen von Windows Active Directory (AD)

1. Folgen Sie den Schritten, die unter Erstellen Sie Ihr AWS verwaltetes AD-Verzeichnis im AWS Directory Service Administration Guide beschrieben sind. Verwenden Sie die VPC, die Sie für dieses Lernprogramm festgelegt haben. In Schritt 3 von Erstellen Ihres von AWS verwalteten AD-Verzeichnis speichern Sie den Benutzernamen und das Kennwort für die Verwendung in einem folgenden Schritt. Beachten Sie auch den vollqualifizierten Domain-Namen für zukünftige Schritte. Sie können den folgenden Schritt ausführen, während das Active Directory erstellt wird.
2. Erstellen Sie ein AWS Secrets Manager Manager-Geheimnis, das Sie in den folgenden Schritten verwenden können. Weitere Informationen finden Sie unter Erste Schritte mit AWS Secrets Manager im AWS Secrets Manager Manager-Benutzerhandbuch.
   a. Öffnen Sie die Secrets Manager-Konsole.
   b. Klicken Sie auf Speichern eines neuen Secrets.
   c. Wählen Sie Andere Art von Secrets aus.
   d. Erstellen Sie für Secret key/value in der ersten Zeile einen Schlüssel **`username`** mit Wert **`admin`**. Klicken Sie auf das Symbol + Zeile hinzufügen.
   e. Erstellen Sie in der neuen Zeile einen Schlüssel **`password`**. Geben Sie als Wert das Passwort ein, das Sie in Schritt 3 von Create Your AWS Managed AD Directory eingegeben haben.
   f. Klicken Sie auf die Schaltfläche Weiter.
   g. Geben Sie einen Namen und eine Beschreibung des Secrets ein. Klicken Sie auf Weiter.
   h. Klicken Sie auf Weiter. Klicken Sie auf Speichern.
   i. Klicken Sie auf der Seite der Secrets auf das Secret, das Sie gerade erstellt haben.
   j. Speichern Sie den ARN des neuen Secrets für die Verwendung in den folgenden Schritten.
   k. Sie können mit dem nächsten Schritt fortfahren, während Ihr Active Directory erstellt wird.

Schritt 3: Überprüfen und aktualisieren Sie Ihre Sicherheitsgruppe

In diesem Schritt überprüfen und aktualisieren Sie die Regeln für die Sicherheitsgruppe, die Sie verwenden. Sie können die Standardsicherheitsgruppe, die für Ihre VPC erstellt wurde, verwenden.

Überprüfen und aktualisieren Sie die Sicherheitsgruppe.

Sie müssen Ihre Sicherheitsgruppe erstellen oder bearbeiten, um Daten von und an die Ports zu senden, die unter Amazon VPC-Sicherheitsgruppen im Benutzerhandbuch für FSx for Windows File Server beschrieben sind. Sie können dies tun, indem Sie die Sicherheitsgruppenregel erstellen, die in der ersten Zeile der folgenden Tabelle mit eingehenden Regeln angezeigt wird. Diese Regel lässt eingehenden Datenverkehr von Netzwerkschnittstellen (und den zugehörigen Instances) zu, die derselben Sicherheitsgruppe zugewiesen sind. Alle von Ihnen erstellten Cloud-Ressourcen befinden sich in derselben VPC und derselben Sicherheitsgruppe zugeordnet. Daher ermöglicht diese Regel, dass Datenverkehr nach Bedarf an und vom FSx for Windows File Server Dateisystem, Active Directory und ECS-Instance gesendet werden kann. Mit den anderen eingehenden Regeln kann der Datenverkehr die Website bereitstellen und den RDP-Zugriff für die Verbindung mit Ihrer ECS-Instance herstellen.

Die folgende Tabelle zeigt, welche Sicherheitsgruppeneingangsregeln für dieses Lernprogramm erforderlich sind.

Typ	Protocol (Protokoll)	Port-Bereich	Quelle
Gesamter Datenverkehr	Alle	Alle	*sg-securitygroup*
HTTPS	TCP	443	0.0.0.0/0
RDP	TCP	3389	Ihre Laptop-IP-Adresse

Die folgende Tabelle zeigt, welche Regeln für ausgehende Sicherheitsgruppen für dieses Lernprogramm erforderlich sind.

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Gesamter Datenverkehr	Alle	Alle	0.0.0.0/0

1. Öffnen Sie EC2-Konsole und wählen Sie dann Sicherheitsgruppen vom Menü links.
2. Aktivieren Sie in der Liste der jetzt angezeigten Sicherheitsgruppen das Kontrollkästchen links neben der Sicherheitsgruppe, die Sie für dieses Tutorial verwenden.

   Die Details Ihrer Sicherheitsgruppe werden angezeigt.
3. Bearbeiten Sie die ein- und ausgehenden Regeln, indem Sie die Registerkarten Inbound rules (Regeln für eingehenden Datenverkehr) oder Outbound rules (Regeln für ausgehenden Datenverkehr) auswählen und die Schaltflächen Edit inbound rules (Bearbeiten von Regeln für eingehenden Datenverkehr) oder Edit outbound rules (Bearbeiten von Regeln für ausgehenden Datenverkehr) auswählen. Bearbeiten Sie die Regeln so, dass sie mit den in den vorhergehenden Tabellen angezeigten Regeln übereinstimmen. Nachdem Sie später in diesem Lernprogramm Ihre EC2-Instance erstellt haben, bearbeiten Sie die RDP-Quelle für eingehende Regeln mit der öffentlichen IP-Adresse Ihrer EC2-Instance, wie unter Herstellen einer Verbindung mit Ihrer Windows-Instance aus dem Amazon EC2 Benutzerhandbuch für Windows-Instances beschrieben.

Schritt 4: Erstellen eines Amazon FSx for Windows File Server-Dateisystems

Nachdem Ihre Sicherheitsgruppe überprüft und aktualisiert wurde und Ihr Active Directory erstellt wurde und sich im aktiven Status befindet, erstellen Sie das Dateisystem FSx for Windows File Server in derselben VPC wie Ihr Active Directory. Führen Sie die folgenden Schritte aus, um ein FSx for Windows File Server Dateisystem für Ihre Windows-Aufgaben zu erstellen.

Erstellen Sie Ihr erstes Dateisystem.

1. Öffnen Sie die Amazon FSx-Konsole.
2. Klicken Sie auf dem Dashboard auf Create file system (Dateisystem erstellen), um den Erstellungsassistenten für Dateisysteme zu starten.
3. Wählen Sie auf der Seite Wählen Sie den Dateisystemtyp FSx for Windows File Server und wählen Sie dann Weiter aus. Die Seite Create file system (Dateisystem erstellen) wird angezeigt.
4. Geben Sie im Abschnitt Details zum Dateisystem einen Namen für Ihr Dateisystem an. Die Benennung Ihrer Dateisysteme erleichtert die Suche und Verwaltung Ihrer Dateisysteme. Sie können bis zu 256 Unicode-Zeichen verwenden. Erlaubt sind Buchstaben, Zahlen, Leerzeichen sowie die Sonderzeichen (+). Minuszeichen (-), Gleichheitszeichen (=), Punkt (.), Unterstrich (_), Doppelpunkt (:), und Schrägstrich (/).

5. Wählen Sie für Deployment type (Bereitstellungstyp) Single-AZ, um ein Dateisystem bereitzustellen, das in einer einzelnen Availability Zone bereitgestellt wird. Single-AZ 2 ist die neueste Generation einzelner Availability Zone-Dateisysteme und unterstützt SSD- und HDD-Speicher.
6. Wählen Sie unter Storage type (Speichertyp) die Option HDD aus.
7. Geben Sie für Speicherkapazität die Mindestspeicherkapazität ein.
8. Behalten Sie die Durchsatzkapazität auf ihrer Standardeinstellung.
9. Wählen Sie im Bereich Netzwerk und Sicherheit dieselbe Amazon VPC aus, die Sie für Ihr AWS Directory Service Verzeichnis ausgewählt haben.
10. Wählen Sie für VPC-Sicherheitsgruppen die Sicherheitsgruppe aus, die Sie in Schritt 3: Überprüfen und Aktualisieren der Sicherheitsgruppe geprüft haben.
11. Wählen Sie für Windows-Authentifizierung Von AWS verwaltetes Microsoft Active Directory und wählen Sie dann Ihr AWS Directory Service -Verzeichnis aus der Liste.
12. Behalten Sie für Verschlüsselung die Standardeinstellung Verschlüsselungsschlüssel von aws/fsx (Standard) bei.
13. Behalten Sie die Standardeinstellungen für Einstellungen für Wartung bei.
14. Klicken Sie auf die Schaltfläche Weiter.
15. Prüfen Sie die Dateisystemkonfiguration, die auf der Seite Create File System (Dateisystem erstellen) angezeigt wird. Beachten Sie, welche Dateisystemeinstellungen Sie nach dem Erstellen des Dateisystems ändern können. Wählen Sie Create file system (Dateisystem erstellen) aus.
16. Notieren Sie sich die File system ID (Dateisystem-ID). Sie werden sie in einem späteren Schritt verwenden müssen.

    Sie können mit den nächsten Schritten fortfahren, um einen Cluster und eine EC2-Instance zu erstellen, während das Dateisystem FSx for Windows File Server erstellt wird.

Schritt 5: Erstellen eines Amazon-ECS-Clusters

Einen Cluster mit der Amazon-ECS-Konsole erstellen

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.

4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Geben Sie unter Cluster-Konfiguration für Cluster-Name windows-fsx-cluster ein.
6. Erweitern Sie Infrastruktur, löschen Sie AWS Fargate (serverlos) und wählen Sie dann Amazon EC2 EC2-Instances aus.

    - Um eine Auto-Scaling-Gruppe zu erstellen, wählen Sie in der Auto-Scaling-Gruppe (ASG) Create new group (Neue Gruppe erstellen) und geben Sie dann die folgenden Details zur Gruppe an:

        - Wählen Sie für Betriebssystem/Architektur Windows Server 2019 Core aus.
        - Wählen Sie unter EC2-Instance-Typ t2.medium oder t2.micro.
7. Wählen Sie Erstellen.

Schritt 6: Eine Amazon-ECS-optimierte Amazon-EC2-Instance erstellen

Erstellen Sie eine Windows-Container-Instance für Amazon ECS.

So erstellen Sie eine Amazon-ECS-Instance

1. Verwenden Sie den `aws ssm get-parameters`-Befehl, um den AMI-Namen für die Region abzurufen, in der Ihre VPC gehostet wird. Weitere Informationen finden Sie unter Abrufen von Amazon-ECS-optimierten AMI-Metadaten.
2. Verwenden Sie die Amazon-EC2-Konsole, um die Instance zu starten.

    a. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.

    b. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.

    c. Wählen Sie im EC2-Dashboard Launch Instance (Instance starten) aus.

    d. Geben Sie für Name einen eindeutigen Namen ein.

    e. Für Anwendungs- und Betriebssystem-Images (Amazon Machine Image) geben Sie in das Suchfeld den Namen des AMI ein, das Sie abgerufen haben.

    f. Wählen Sie unter Instance-Typ t2.medium oder t2.micro.

    g. Wählen Sie für Key pair (login) (Schlüsselpaar (Anmeldung)) ein Schlüsselpaar aus. Wenn Sie kein Schlüsselpaar angeben, werden Sie

    h. Wählen Sie unter Netzwerkeinstellungen für VPC und Subnetz Ihre VPC und ein öffentliches Subnetz.

i. Wählen Sie unter Network settings (Netzwerkeinstellungen) für Security group (Sicherheitsgruppe) eine vorhandene Sicherheitsgruppe aus oder erstellen Sie eine neue. Stellen Sie sicher, dass für die von Ihnen gewählte Sicherheitsgruppe die Regeln für eingehenden und ausgehenden Datenverkehr in Voraussetzungen für das Tutorial definiert sind

j. Wählen Sie unter Network settings (Netzwerkeinstellungen), für Auto-assign Public IP (Öffentliche IP automatisch zuweisen), die Option Enable (Aktivieren) aus.

k. Erweitern Sie Erweiterte Details und wählen Sie dann für Domain-Beitrittsverzeichnis die ID des Active Directory, das Sie erstellt haben. Diese Options-Domain tritt Ihrem AD bei, wenn die EC2-Instance gestartet wird.

l. Wählen Sie unter Erweiterte Details für das IAM-Instance-Profil die Option ecs aus. InstanceRole

m. Konfigurieren Sie Ihre Amazon-ECS-Container-Instance mit den folgenden Benutzerdaten. Fügen Sie unter Advanced Details (Erweiterte Details) das folgende Skript in das Feld User data (Benutzerdaten) ein und ersetzen Sie *cluster_name* durch den Namen Ihres Clusters.

```
<powershell>
Initialize-ECSAgent -Cluster windows-fsx-cluster -EnableTaskIAMRole
</powershell>
```

n. Wenn Sie bereit sind, wählen Sie das Bestätigungsfeld und danach Launch Instances aus.

o. Auf einer Bestätigungsseite wird Ihnen mitgeteilt, dass die Instance gestartet wird. Wählen Sie View Instances aus, um die Bestätigungsseite zu schließen und zur Konsole zurückzukehren.

3. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.

4. Wählen Sie im Navigationsbereich Cluster und dann windows-fsx-cluster.

5. Wählen Sie die Registerkarte Infrastruktur und vergewissern Sie sich, dass Ihre Instance im Cluster windows-fsx-cluster registriert ist.

Schritt 7: Registrieren einer Windows-Aufgabendefinition

Bevor Sie in Ihrem Amazon-ECS-Cluster Windows-Container ausführen können, müssen Sie eine Aufgabendefinition registrieren. Das folgende Beispiel für eine Aufgabendefinition zeigt eine einfache Webseite. Die Aufgabe startet zwei Container, die Zugriff auf das FSx Dateisystem haben. Der erste

Container schreibt eine HTML-Datei in das Dateisystem. Der zweite Container lädt die HTML-Datei aus dem Dateisystem herunter und bedient die Webseite.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
4. Ersetzen Sie im Feld JSON-Editor die Werte für Ihre Aufgabenausführungsrolle und die Details zu Ihrem FSx-Dateisystem und wählen Sie dann Speichern.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType
 file -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body
 {margin-top: 40px; background-color: #333;} </style> </head><body> <div
 style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1> <h2>It
 Works!</h2> <p>You are using Amazon FSx for Windows File Server file system for
 persistent container storage.</p>' -Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
```

```
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force;
 Start-Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -
Destination C:\\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe
 w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

Schritt 8: Eine Aufgabe ausführen und die Ergebnisse anzeigen

Stellen Sie vor dem Ausführen der Aufgabe sicher, dass der Status Ihres FSx for Windows File Server Dateisystems Verfügbar ist. Nachdem sie verfügbar ist, können Sie eine Aufgabe mithilfe der Aufgabendefinition ausführen, die Sie erstellt haben. Die Aufgabe beginnt mit dem Erstellen von Containern, die eine HTML-Datei zwischen ihnen mithilfe des Dateisystems mischen. Nach dem Shuffle bedient ein Webserver die einfache HTML-Seite.

> **Note**
>
> Möglicherweise können Sie keine Verbindung zu der Website innerhalb eines VPN herstellen.

Führen Sie eine Aufgabe aus und zeigen die Ergebnisse mit der Amazon-ECS-Konsole an.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Cluster und dann windows-fsx-cluster.
3. Wählen Sie die Registerkarte Aufgaben und dann Neue Aufgabe ausführen.
4. Wählen Sie unter Launch Type EC2 aus.
5. Wählen Sie unter Bereitstellungskonfiguration für Aufgabendefinition den Eintrag fsx-windows und wählen Sie dann Erstellen.
6. Wenn Ihr Aufgabenstatus WIRD AUSGEFÜHRT ist, wählen Sie die Aufgaben-ID aus.
7. Wählen Sie unter Container, wenn der Status von Container1 auf ANGEHALTEN steht, Container2, um die Details des Containers anzuzeigen.
8. Wählen Sie unter Container-Details für Container2 die Option Netzwerkbindungen aus und klicken Sie dann auf die externe IP-Adresse, die dem Container zugeordnet ist. Ihr Browser wird geöffnet und die folgende Meldung wird angezeigt.

```
Amazon ECS Sample App
It Works!
You are using Amazon FSx for Windows File Server file system for persistent
 container storage.
```

> **Note**
>
> Es kann einige Minuten dauern, bis die Meldung angezeigt wird. Wenn Sie diese Meldung nach einigen Minuten nicht angezeigt wird, stellen Sie sicher, dass Sie nicht in

einem VPN ausgeführt wird, und stellen Sie sicher, dass die Sicherheitsgruppe für Ihre Container-Instance eingehenden Netzwerk-HTTP-Datenverkehr auf Port 443 zulässt.

Schritt 9: Bereinigen

> **Note**
>
> Das Löschen des FSx for Windows File Server-Dateisystems oder des AD dauert 20 bis 45 Minuten. Warten Sie, bis die FSx for Windows File Server-Dateisystemlöschvorgänge abgeschlossen sind, bevor Sie die AD Löschvorgänge starten.

Löschen eines Dateisystems von FSx für Windows File Server.

1. Öffnen Sie die Amazon FSx-Konsole
2. Wählen Sie das Optionsfeld links neben dem Dateisystem von FSx für Windows File Server, das Sie gerade erstellt haben.
3. Wählen Sie Aktionen.
4. Wählen Sie Dateisystem löschen.

Löschen Sie die AD.

1. Öffnen Sie die AWS Directory Service -Konsole.
2. Wählen Sie das Optionsfeld links neben dem gerade erstellten AD.
3. Wählen Sie Aktionen.
4. Wählen Sie Löschen eines Verzeichnisses aus.

Löschen Sie den Cluster.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Cluster und dann fsx-windows-cluster.
3. Wählen Sie Delete cluster (Cluster löschen) aus.
4. Geben Sie die Phrase ein und wählen Sie dann Löschen.

Beenden Sie die EC2-Instance.

1. Öffnen Sie die Amazon EC2-Konsole.
2. Wählen Sie im linken Menü die Option Instances aus.
3. Markieren Sie das Kästchen links neben der EC2-Instance, die Sie erstellt haben.
4. Klicken Sie auf Instance-Zustand, Beenden der Instance.

Löschen Sie das Secret.

1. Öffnen Sie die Secrets Manager-Konsole.
2. Wählen Sie das Secret aus, das Sie für diesen Walkthrough erstellt haben.
3. Klicken Sie auf Aktionen.
4. Wählen Sie Secret löschen aus.

Verwenden Sie Docker-Volumes mit Amazon ECS

Bei der Verwendung von Docker-Volumes kann der integrierte `local`-Treiber oder ein Drittanbieter-Volume-Treiber verwendet werden. Docker-Volumes werden von Docker verwaltet und es wird ein Verzeichnis in `/var/lib/docker/volumes` auf der Container-Instance erstellt, die die Volume-Daten enthält.

Um Docker-Volumes zu verwenden, geben Sie eine `dockerVolumeConfiguration` in Ihrer Aufgabendefinition an. Weitere Informationen finden Sie unter Verwenden von Volumes.

Einige häufige Anwendungsfälle für Docker-Volumes sind folgende:

- Das Bereitstellen von persistenten Daten-Volumes für die Nutzung mit Containern
- Die gemeinsame Verwendung eines definierten Daten-Volumes an unterschiedlichen Orten auf verschiedenen Containern auf derselben Container-Instance
- Das Definieren eines leeren, nicht persistenten Daten-Volumes und dessen Mounten auf mehreren Containern innerhalb der gleichen Aufgabe
- Das Bereitstellen eines Daten-Volumes für Ihre Aufgabe, die von einem Drittanbieter-Treiber verwaltet wird

Überlegungen zur Verwendung von Docker-Volumes

Bei der Verwendung von Docker-Volumes sollte Folgendes berücksichtigt werden:

- Docker-Volumes werden nur unterstützt, wenn der EC2-Launchtyp oder externe Instances verwendet werden.
- Windows-Container unterstützen nur die Verwendung des `local`-Treibers.
- Wenn ein Drittanbieter-Treiber verwendet wird, stellen Sie sicher, dass dieser installiert und auf der Container-Instance aktiv ist, bevor der Container-Agent gestartet wird. Wenn der Drittanbieter-Treiber nicht aktiv ist, bevor Sie den Agenten starten, können Sie den Container-Agenten neu starten, indem Sie einen der folgenden Befehle verwenden:
  - Für das Amazon-ECS-optimierte Amazon Linux 2-AMI:

    ```
    sudo systemctl restart ecs
    ```

  - Für das Amazon-ECS-optimierte Amazon Linux AMI:

    ```
    sudo stop ecs && sudo start ecs
    ```

Geben Sie ein Docker-Volume in einer Amazon ECS-Aufgabendefinition an

Bevor Ihre Container Daten-Volumes verwenden können, müssen Sie das Volume und die Konfigurationen der Mountingpunkte in Ihrer Aufgabendefinition angeben. Dieser Abschnitt beschreibt die Volume-Konfiguration für einen Container. Für Aufgaben, die ein Docker-Volume verwenden, geben Sie eine `dockerVolumeConfiguration` an. Für Aufgaben, die ein Bind-Mount-Host-Volume verwenden, geben Sie einen `host` und optionalen `sourcePath` an.

Das folgende JSON-Codefragment der Aufgabendefinition zeigt die Syntax für die Objekte `volumes` und `mountPoints` für einen Container.

```
{
    "containerDefinitions": [
        {
            "mountPoints": [
                {
                    "sourceVolume": "string",
                    "containerPath": "/path/to/mount_volume",
                    "readOnly": boolean
                }
```

```
            ]
        }
    ],
    "volumes": [
        {
            "name": "string",
            "dockerVolumeConfiguration": {
                "scope": "string",
                "autoprovision": boolean,
                "driver": "string",
                "driverOpts": {
                    "key": "value"
                },
                "labels": {
                    "key": "value"
                }
            }
        }
    ]
}
```

`name`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Volumes. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Zahlen, Bindestriche () und Unterstriche (-) sind zulässig. _ Auf diesen Namen wird im `sourceVolume` Parameter des Container-Definitionsobjekts verwiesen. `mountPoints`

`dockerVolumeConfiguration`

Typ: DockerVolumeKonfigurationsobjekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von Docker-Volumes angegeben. Docker-Volumes werden nur unterstützt, wenn Aufgaben auf EC2-Instances ausgeführt werden. Windows-Container unterstützen nur die Verwendung des `local` Treibers. Um Bind-Mounts zu verwenden, geben Sie stattdessen einen `host` an.

`scope`

Typ: Zeichenfolge

Zulässige Werte: `task` | `shared`

Erforderlich: Nein

Der Bereich für das Docker-Volume, der den Lebenszyklus bestimmt. Docker-Volumes, die auf eine `task` beschränkt sind, werden automatisch beim Starten der Aufgabe bereitgestellt und beim Stoppen dieser vernichtet. Docker-Volumes, die als `shared` angewendet werden, bleiben erhalten, nachdem die Aufgabe gestoppt wird.

`autoprovision`

Typ: Boolesch

Standardwert: `false`

Erforderlich: Nein

Wenn dieser Wert `true` lautet, wird das Docker-Volume erstellt, wenn es nicht bereits vorhanden ist. Dieses Feld wird nur verwendet, wenn `scope` der`shared`. Wenn der `scope` ist`task`, muss dieser Parameter entweder weggelassen oder auf gesetzt werden`false`.

`driver`

Typ: Zeichenfolge

Erforderlich: Nein

Der zu verwendende Docker-Volume-Treiber. Der Treiberwert muss mit dem von Docker bereitgestellten Treibernamen übereinstimmen, da dieser Name für die Aufgabenplatzierung verwendet wird. Wenn der Treiber mithilfe der Docker-Plug-in-CLI installiert wurde, verwenden Sie ihn, `docker plugin ls` um den Treibernamen von Ihrer Container-Instance abzurufen. Wenn der Treiber mit einer anderen Methode installiert wurde, verwenden Sie die Docker-Plug-in-Erkennung, um den Treibernamen abzurufen. Weitere Informationen finden Sie unter Docker-Plug-In-Erkennung. Dieser Parameter ist `Driver` im Bereich Create a volume der Docker-Remote-API und der `--driver`-Option zum `docker volume create` zugewiesen.

`driverOpts`

Typ: Zeichenfolge

Erforderlich: Nein

Eine Übersicht mit Treiberspezifischen Optionen für den Docker-Treiber, die durchgespielt werden sollen. Dieser Parameter ist `DriverOpts` im Bereich Create a volume der Docker-Remote-API und der `--opt`-Option zum `docker volume create` zugewiesen.

`labels`

Typ: Zeichenfolge

Erforderlich: Nein

Benutzerdefinierte Metadaten, die Ihrem Docker-Volume hinzugefügt werden sollen. Dieser Parameter ist `Labels` im Bereich Create a volume der Docker-Remote-API und der `--label`-Option zum `docker volume create` zugewiesen.

`mountPoints`

Typ: Objekt-Array

Erforderlich: Nein

Die Bereitstellungspunkte für die Datenvolumes in Ihrem Container. Dieser Parameter ordnet zu `Volumes` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volume` für die docker run zu.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten. Windows-Container können keine Verzeichnisse auf einem anderen Laufwerk mounten, und Bereitstellungspunkte können nicht laufwerksübergreifend verwendet werden. Sie müssen Bereitstellungspunkte angeben, um ein Amazon EBS-Volume direkt an eine Amazon ECS-Aufgabe anzuhängen.

`sourceVolume`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Name des zu mountenden Volumes.

`containerPath`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Pfad im Container, in dem das Volume bereitgestellt werden soll.

`readOnly`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume. Der Standardwert ist `false`.

Beispiele für Docker-Volumes

Um kurzlebigen Speicher für einen Container bereitzustellen, der ein Docker-Volume verwendet

In diesem Beispiel verwendet ein Container ein leeres Datenvolumen, das nach Beendigung der Aufgabe entsorgt wird. Ein Beispielanwendungsfall ist, dass Sie z. B. einen Container besitzen, der während einer Aufgabe auf irgendeinen Scratch-Dateispeicherort zugreifen muss. Diese Aufgabe kann mit einem Docker-Volume gelöst werden.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Datenvolume mit `name`- und `DockerVolumeConfiguration`-Werten. In diesem Beispiel geben wir den Umfang als `task` an, sodass das Volume gelöscht wird, nachdem die Aufgabe angehalten wurde. Zudem wird der `local`-Treiber verwendet.

```
"volumes": [
    {
        "name": "scratch",
        "dockerVolumeConfiguration" : {
            "scope": "task",
            "driver": "local",
            "labels": {
                "scratch": "space"
            }
        }
    }
]
```

2. Definieren Sie im Abschnitt `containerDefinitions` einen Container mit `mountPoints`-Werten, die auf den Namen des definierten Volumes und den Wert `containerPath` verweisen, um das Volume auf den Container zu mounten.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
            {
              "sourceVolume": "scratch",
              "containerPath": "/var/scratch"
            }
        ]
    }
]
```

So stellen Sie persistenten Speicher für einen Container mit einem Docker-Volume bereit

In diesem Beispiel möchten Sie, dass ein freigegebenes Volume für mehrere Container verwendet werden soll und es soll bestehen bleiben, nachdem jede einzelne Aufgabe gestoppt wurde, die es verwendet. Der integrierte `local`-Treiber wird verwendet, damit das Volume weiterhin mit dem Lebenszyklus der Container-Instance verknüpft ist.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Datenvolume mit `name`- und `DockerVolumeConfiguration`-Werten. In diesem Beispiel geben Sie einen `shared`-Geltungsbereich an, damit das Volume bestehen bleibt. Legen Sie die automatische Bereitstellung auf `true` fest, damit das Volume zur Verwendung erstellt wird. Verwenden Sie dann auch den integrierten `local`-Treiber.

```
"volumes": [
    {
        "name": "database",
        "dockerVolumeConfiguration" : {
            "scope": "shared",
            "autoprovision": true,
            "driver": "local",
            "labels": {
                "database": "database_name"
            }
        }
    }
]
```

2. Definieren Sie im Abschnitt `containerDefinitions` einen Container mit `mountPoints`-Werten, die auf den Namen des definierten Volumes und den Wert `containerPath` verweisen, um das Volume auf den Container zu mounten.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    },
    {
      "name": "container-2",
      "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    }
  ]
```

So stellen Sie NFS-persistenten Speicher für einen Container mit einem Docker-Volume bereit

In diesem Beispiel verwendet ein Container ein NFS-Datenvolumen, das beim Start der Aufgabe automatisch gemountet und beim Beenden der Aufgabe automatisch getrennt wird. Dies verwendet den in Docker integrierten `local`-Treiber. Ein Beispiel für einen Anwendungsfall ist, dass Sie möglicherweise über einen lokalen NFS-Speicher verfügen und von einer ECS-Anywhere-Aufgabe aus darauf zugreifen müssen. Dies kann mit einem Docker-Volume mit NFS-Treiberoption erreicht werden.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Datenvolume mit `name`- und `DockerVolumeConfiguration`-Werten. Geben Sie in diesem Beispiel einen `task`-Bereich an, damit das Volume nach Beendigung der Aufgabe getrennt wird. Verwenden Sie den `local`-Treiber und konfigurieren Sie `driverOpts` mit den `type`-, `device`-, and o-Optionen entsprechend. Ersetzen Sie `NFS_SERVER` durch den NFS-Serverendpunkt.

```
"volumes": [
       {
           "name": "NFS",
           "dockerVolumeConfiguration" : {
               "scope": "task",
               "driver": "local",
               "driverOpts": {
                   "type": "nfs",
                   "device": "$NFS_SERVER:/mnt/nfs",
                   "o": "addr=$NFS_SERVER"
               }
           }
       }
   ]
```

2. Definieren Sie im `containerDefinitions`-Abschnitt einen Container mit `mountPoints`-Werten, die auf den Namen des definierten Volumes und den `containerPath`-Wert verweisen, um das Volume im Container bereitzustellen.

```
"containerDefinitions": [
       {
           "name": "container-1",
           "mountPoints": [
               {
                 "sourceVolume": "NFS",
                 "containerPath": "/var/nfsmount"
               }
           ]
       }
   ]
```

Bind-Mounts mit Amazon ECS verwenden

Bei Bind-Mounts wird eine Datei oder ein Verzeichnis auf einem Host, z. B. einer Amazon EC2 EC2-Instance, in einen Container gemountet. Bind-Mounts werden für Aufgaben unterstützt, die auf Fargate und Amazon-EC2-Instances gehostet werden. Bind-Mounts sind an den Lebenszyklus des Containers gebunden, der sie verwendet. Nachdem alle Container, die ein Bind-Mount verwenden, gestoppt wurden, z. B. wenn eine Aufgabe gestoppt wird, werden die Daten entfernt. Bei Aufgaben, die auf Amazon EC2 EC2-Instances gehostet werden, können die Daten an den Lebenszyklus der

Amazon EC2 EC2-Hostinstanz gebunden werden, indem Sie in Ihrer Aufgabendefinition einen `host` optionalen `sourcePath` Wert angeben. Weitere Informationen finden Sie unter Verwenden von Bind-Mounts in der Docker-Dokumentation.

Die folgenden Szenarien sind gängige Anwendungsfälle für Bind-Mounts.

- So stellen Sie ein leeres Datenvolume bereit, das in einem oder mehreren Containern bereitgestellt werden soll.
- So stellen Sie ein Hostdaten-Volume in einem oder mehreren Containern bereit.
- So geben Sie ein Daten-Volume aus einem Quellcontainer für andere Container in derselben Aufgabe frei.
- Um einen Pfad und seinen Inhalt aus einer Dockerdatei für einen oder mehrere Container verfügbar zu machen.

Überlegungen zur Verwendung von Bind-Mounts

Bei der Verwendung von Bind-Mounts sollte Folgendes berücksichtigt werden.

- Standardmäßig erhalten Aufgaben, die auf der AWS Fargate Plattformversion `1.4.0` oder höher (Linux) `1.0.0` oder höher (Windows) gehostet werden, mindestens 20 GiB flüchtigen Speicher für Bind-Mounts. Sie können die Gesamtmenge des flüchtigen Speichers auf maximal 200 GiB erhöhen, indem Sie den `ephemeralStorage` Parameter in Ihrer Aufgabendefinition angeben.
- Um Dateien aus einer Dockerdatei auf einem Datenvolume verfügbar zu machen, wenn eine Aufgabe ausgeführt wird, sucht die Amazon-ECS-Datenebene nach einer `VOLUME`-Richtlinie. Wenn der absolute Pfad, der in der `VOLUME`-Richtlinie angegeben ist der gleiche wie `containerPath` ist, die in der Aufgabendefinition angegeben sind, werden die Daten im `VOLUME`-Richtlinienpfad auf das Datenvolume kopiert. Im folgenden Dockerfile-Beispiel wird eine Datei mit dem Namen `examplefile` im `/var/log/exported`-Verzeichnis auf den Host geschrieben und dann innerhalb des Containers gemountet.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN mkdir -p /var/log/exported
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

Standardmäßig sind die Volume-Berechtigungen auf `0755` und der Besitzer als `root` festgelegt. Sie können diese Berechtigungen in der Dockerfile anpassen. Das folgende Beispiel definiert den Eigentümer des Verzeichnisses als `node`.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install -y shadow-utils && yum clean all
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
RUN touch /var/log/exported/examplefile
USER node
VOLUME ["/var/log/exported"]
```

- Bei Aufgaben, die auf Amazon-EC2-Instances gehostet werden, wenn ein `host`- und `sourcePath`-Wert nicht angegeben ist, verwaltet der Docker-Daemon das Bind-Mount für Sie. Wenn keine Container auf dieses Bind-Mount verweisen, löscht der Aufgabenbereinigungsservice des Amazon-ECS-Container-Agenten es letztendlich. Standardmäßig geschieht dies drei Stunden nach dem Stoppen des Containers. Die Zeitspanne kann aber mit der Agenten-Variable `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` konfiguriert werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten. Wenn Sie diese Daten länger als den Lebenszyklus des Containers beibehalten möchten, geben Sie für die Bind-Bindungsbereitstellung einen `sourcePath`-Wert an.

Geben Sie einen Bind-Mount in einer Amazon ECS-Aufgabendefinition an

Für Amazon ECS-Aufgaben, die entweder auf Fargate- oder Amazon EC2 EC2-Instances gehostet werden, zeigt das folgende JSON-Snippet mit der Aufgabendefinition die Syntax für die`volumes`, `mountPoints` ,- und `ephemeralStorage` Objekte für eine Aufgabendefinition.

```
{
   "family": "",
   ...
   "containerDefinitions" : [
      {
         "mountPoints" : [
            {
               "containerPath" : "/path/to/mount_volume",
               "sourceVolume" : "string"
            }
          ],
```

```
            "name" : "string"
        }
    ],
    ...
    "volumes" : [
        {
            "name" : "string"
        }
    ],
    "ephemeralStorage": {
    "sizeInGiB": integer
    }
}
```

Für Amazon-ECS-Aufgaben, die auf Amazon-EC2-Instances gehostet werden, können Sie den optionalen `host`-Parameter und `sourcePath` nutzen, wenn Sie die Details des Aufgaben-Volumes angeben. Sind diese angegeben, wird das Bind-Mount an den Lebenszyklus der Aufgabe und nicht an den Container gebunden.

```
"volumes" : [
    {
        "host" : {
            "sourcePath" : "string"
        },
        "name" : "string"
    }
]
```

Im Folgenden finden Sie weitere detaillierte Beschreibungen der einzelnen Aufgabendefinitionsparameter.

`name`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Volumes. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Zahlen, Bindestriche () und Unterstriche (`-`) sind zulässig. _ Auf diesen Namen wird im `sourceVolume` Parameter des Container-Definitionsobjekts verwiesen. `mountPoints`

`host`

Erforderlich: Nein

Der `host`-Parameter wird verwendet, um den Lebenszyklus des Bind-Mounts an die Amazon-EC2-Host-Instance und nicht an die Aufgabe zu binden und dort zu speichern. Wenn der Parameter `host` leer ist, weist der Docker-Daemon einen Hostpfad für Ihr Daten-Volume zu, es wird aber nicht gewährleistet, dass die Daten beibehalten werden, nachdem die damit verknüpften Container nicht mehr ausgeführt werden.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten.

> **Note**
>
> Der `sourcePath` Parameter wird nur unterstützt, wenn Aufgaben verwendet werden, die auf Amazon EC2 EC2-Instances gehostet werden.

`sourcePath`

Typ: Zeichenfolge

Erforderlich: Nein

Wenn der Parameter `host` verwendet wird, geben Sie einen `sourcePath` an, um den Pfad auf der Amazon-EC2-Host-Instance zu deklarieren, die dem Container bereitgestellt wird. Wenn dieser Parameter leer ist, weist der Docker-Daemon einen Host-Pfad für Sie zu. Wenn der Parameter `host` den Speicherort `sourcePath` enthält, bleibt das Daten-Volume an der angegebenen Position der Amazon-EC2-Host-Instance erhalten, bis Sie es manuell löschen. Wenn der Wert `sourcePath` auf der Amazon-EC2-Host-Instance nicht vorhanden ist, wird er vom Docker-Daemon erstellt. Wenn der Speicherort nicht vorhanden ist, wird der Inhalt des Quellpfadordners exportiert.

`mountPoints`

Typ: Objekt-Array

Erforderlich: Nein

Die Bereitstellungspunkte für die Datenvolumes in Ihrem Container. Dieser Parameter ordnet zu `Volumes` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volume` für die docker run zu.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten. Windows-Container können keine Verzeichnisse auf einem anderen Laufwerk mounten, und Bereitstellungspunkte können nicht laufwerksübergreifend verwendet werden. Sie müssen Bereitstellungspunkte angeben, um ein Amazon EBS-Volume direkt an eine Amazon ECS-Aufgabe anzuhängen.

`sourceVolume`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Name des zu mountenden Volumes.

`containerPath`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Pfad im Container, in dem das Volume bereitgestellt werden soll.

`readOnly`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume. Der Standardwert ist `false`.

`ephemeralStorage`

Typ: Objekt

Erforderlich: Nein

Die Menge des flüchtigen Speichers, der für die Aufgabe zugewiesen werden soll. Dieser Parameter wird verwendet, um die Gesamtmenge des verfügbaren kurzlebigen Speichers für Aufgaben, die auf einer Plattformversion `1.4.0` oder höher (Linux) `1.0.0` oder höher (Windows) gehostet werden, AWS Fargate über die Standardmenge hinaus zu erweitern.

Sie können die Copilot-CLI, das AWS SDK oder die CLI verwenden CloudFormation, um kurzlebigen Speicher für einen Bind-Mount anzugeben.

Bind-Mount-Beispiele

So weisen Sie einer Fargate-Aufgabe einen erhöhten flüchtigen Speicher zu

Für Amazon-ECS-Aufgaben, die auf Fargate mit Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` (Windows) gehostet werden können Sie mehr als die Standardmenge des flüchtigen Speichers für die zu verwendenden Container in Ihrer Aufgabe zuweisen. Dieses Beispiel kann in die anderen Beispiele integriert werden, um flüchtigen Speicher für Ihre Fargate-Aufgaben zuzuweisen.

- Definieren Sie in der Aufgabendefinition ein `ephemeralStorage`-Objekt. `sizeInGiB` muss eine Ganzzahl zwischen den Werten von `21` und `200` sein und wird in GiB ausgedrückt.

```
"ephemeralStorage": {
    "sizeInGiB": integer
}
```

So stellen Sie ein leeres Datenvolume für einen oder mehrere Container bereit

In einigen Fällen möchten Sie den Containern in einer Aufgabe einen Scratchspace bereitstellen. Möglicherweise besitzen Sie z. B. zwei Datenbank-Container, die während einer Aufgabe auf denselben Scratch-Dateispeicherort zugreifen müssen. Dies kann mit einem Bind-Mount erreicht werden.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Bind-Mount mit dem Namen `database_scratch`.

```
  "volumes": [
    {
      "name": "database_scratch"
    }
  ]
```

2. Erstellen Sie im Abschnitt `containerDefinitions` die Datenbank-Containerdefinitionen, damit sie das Volume mounten.

```
"containerDefinitions": [
```

```
    {
      "name": "database1",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    },
    {
      "name": "database2",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    }
  ]
```

So stellen Sie einen Pfad und seinen Inhalt in einer Dockerdatei einem Container zur Verfügung

In diesem Beispiel haben Sie eine Dockerdatei, die Daten schreibt, die Sie in einem Container mounten möchten. Dieses Beispiel funktioniert für Aufgaben, die auf Fargate oder Amazon-EC2-Instances gehostet werden.

1. Erstellen Sie eine Docker-Datei. Im folgenden Beispiel wird das öffentliche Amazon Linux 2-Container-Image verwendet und eine Datei mit dem Namen `examplefile` im `/var/log/exported`-Verzeichnis, das wir innerhalb des Containers einhängen möchten. Die `VOLUME`-Direktive sollte einen absoluten Pfad angeben.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN mkdir -p /var/log/exported
   ```

```
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

Standardmäßig sind die Volume-Berechtigungen auf `0755` und der Besitzer als `root` festgelegt. Diese Berechtigungen können in der Dockerdatei geändert werden. Im folgenden Beispiel wird der Besitzer des `/var/log/exported`-Verzeichnis auf `node` festgelegt.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
RUN yum install -y shadow-utils && yum clean all
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
USER node
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

2. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Volume mit dem Namen `application_logs`.

```
"volumes": [
  {
    "name": "application_logs"
  }
]
```

3. Erstellen Sie im Abschnitt `containerDefinitions` die Containerdefinitionen der Anwendung, damit sie den Speicher mounten. Der `containerPath`-Wert muss mit dem absoluten Pfad übereinstimmen, der in der `VOLUME`-Richtlinie aus der Dockerfile angegeben ist.

```
"containerDefinitions": [
  {
    "name": "application1",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
      }
    ]
  },
```

```
    {
      "name": "application2",
      "image": "my-repo/application",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "application_logs",
          "containerPath": "/var/log/exported"
        }
      ]
    }
  ]
```

So stellen Sie ein leeres Datenvolume für einen Container bereit, der mit dem Lebenszyklus der Host-Amazon-EC2-Instance verknüpft ist

Bei Aufgaben, die auf Amazon-EC2-Instances gehostet werden, können Sie Bind-Mounts verwenden und die Daten an den Lebenszyklus der Host-Amazon-EC2-Instance binden. Das geschieht mithilfe des `host`-Parameters und durch Angeben eines `sourcePath`-Werts. Alle Dateien, die im `sourcePath` vorhanden sind, werden den Containern im `containerPath`-Wert angezeigt. Alle Dateien, die in den `containerPath`-Wert geschrieben werden, werden in den `sourcePath`-Wert auf der Host-Amazon-EC2-Instance geschrieben.

> **Important**
>
> Amazon ECS synchronisiert Ihren Speicher nicht über Amazon-EC2-Instances hinweg. Aufgaben, die persistenten Speicher nutzen, können auf eine beliebige Amazon-EC2-Instance in Ihrem Cluster mit verfügbarer Kapazität platziert werden. Wenn Ihre Aufgaben nach dem Stoppen und Neustarten persistenten Speicher benötigen, geben Sie beim Start der Aufgabe immer dieselbe Amazon EC2 EC2-Instance mit dem Befehl AWS CLI start-task an. Sie können Amazon EFS Volumes auch für persistenten Speicher verwenden. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Bind-Mount mit `name`- und `sourcePath`-Werten. Im folgenden Beispiel enthält die Host-Amazon-EC2-Instance Daten unter `/ecs/webdata`, die Sie im Container montieren möchten.

```
"volumes": [
  {
    "name": "webdata",
    "host": {
      "sourcePath": "/ecs/webdata"
    }
  }
]
```

2. Definieren Sie im Abschnitt `containerDefinitions` einen Container mit einem `mountPoints`-Wert, der auf den Namen des definierten Bind-Mounts verweist, und mit dem `containerPath`-Wert, auf dem das Bind-Mount auf dem Container gemountet werden soll.

```
"containerDefinitions": [
  {
    "name": "web",
    "image": "nginx",
    "cpu": 99,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webdata",
        "containerPath": "/usr/share/nginx/html"
      }
    ]
  }
]
```

So mounten Sie ein definiertes Volume auf mehreren Containern an verschiedenen Standorten

Sie können ein Datenvolume in einer Aufgabendefinition definieren und das Volume an verschiedenen Speicherorten auf verschiedenen Containern mounten. Angenommen, Ihr Host-Container besitzt einen Website-Datenordner in `/data/webroot`. Möglicherweise möchten Sie

dieses Datenvolume schreibgeschützt auf zwei verschiedenen Webservern mounten, die über verschiedene Basisverzeichnisse verfügen.

1. Definieren Sie im Abschnitt `volumes` der Aufgabendefinition ein Datenvolume mit dem Namen `webroot` und dem Quellpfad `/data/webroot`.

```
"volumes": [
  {
    "name": "webroot",
    "host": {
      "sourcePath": "/data/webroot"
    }
  }
]
```

2. Definieren Sie im Abschnitt `containerDefinitions` für jeden Webserver einen Container mit `mountPoints`-Werten, die das `webroot`-Volume dem `containerPath`-Wert zuordnen, der auf das Basisverzeichnis des betreffenden Containers verweist.

```
"containerDefinitions": [
  {
    "name": "web-server-1",
    "image": "my-repo/ubuntu-apache",
    "cpu": 100,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webroot",
        "containerPath": "/var/www/html",
        "readOnly": true
      }
    ]
  },
  {
    "name": "web-server-2",
```

```
      "image": "my-repo/sles11-apache",
      "cpu": 100,
      "memory": 100,
      "portMappings": [
        {
          "containerPort": 8080,
          "hostPort": 8080
        }
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/srv/www/htdocs",
          "readOnly": true
        }
      ]
    }
  ]
```

So mounten Sie mit **`volumesFrom`** Volumes aus einem anderen Container

Für Aufgaben, die auf Amazon-EC2-Instances gehostet werden, können Sie ein oder mehrere Volumes in einem Container definieren und dann den `volumesFrom`-Parameter in einer anderen Containerdefinition innerhalb derselben Aufgabe verwenden, um eine Bindungsbereitstellung für alle Volumes von `sourceContainer` an ihrem ursprünglich definierten Bindungsbereitstellungspunkt herzustellen. Der Parameter `volumesFrom` gilt für alle Volumes, die in der Aufgabendefinition definiert sind, und für solche Volumes, die im Image mit einer Dockerfile integriert sind.

1. (Optional) Um ein Volume freizugeben, das in ein Image integriert ist, verwenden Sie die `VOLUME`-Anweisung in der Dockerfile. Im folgenden Beispiel verwendet die Dockerfile ein `httpd`-Image, fügt ein Volume hinzu und mountet es unter `dockerfile_volume` im Stammverzeichnis von Apache. Es ist der Ordner, der vom `httpd`-Webserver verwendet wird.

   ```
   FROM httpd
   VOLUME ["/usr/local/apache2/htdocs/dockerfile_volume"]
   ```

   Sie können ein Image mit dieser Dockerfile erstellen und direkt in ein Repository, wie z. B. den Docker-Hub, übertragen und in Ihrer Aufgabendefinition verwenden. Das `my-repo/`

`httpd_dockerfile_volume` Beispiel-Image, das in den folgenden Schritten verwendet wird, wurde mit dem vorherigen Dockerfile erstellt.

2. Erstellen Sie eine Aufgabendefinition, die Ihre Volumes und Mountingpunkte für die Container definiert. In diesem `volumes`-Beispielabschnitt erstellen Sie ein leeres Volume namens `empty`, das vom Docker-Daemon verwaltet wird. Auch ein Host-Volume namens `host_etc` ist definiert. Es exportiert den `/etc`-Ordner auf der Host-Container-Instance.

```
{
  "family": "test-volumes-from",
  "volumes": [
    {
      "name": "empty",
      "host": {}
    },
    {
      "name": "host_etc",
      "host": {
        "sourcePath": "/etc"
      }
    }
  ],
```

Erstellen Sie im Abschnitt der Containerdefinitionen einen Container, der die zuvor definierten Volumes mountet. In diesem Beispiel mountet der `web`-Container die Volumes `empty` und `host_etc`. Dies ist der Container, der das mit einem Volume in der Dockerfile erstellte Image verwendet.

```
"containerDefinitions": [
    {
      "name": "web",
      "image": "my-repo/httpd_dockerfile_volume",
      "cpu": 100,
      "memory": 500,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "mountPoints": [
        {
```

```
          "sourceVolume": "empty",
          "containerPath": "/usr/local/apache2/htdocs/empty_volume"
        },
        {
          "sourceVolume": "host_etc",
          "containerPath": "/usr/local/apache2/htdocs/host_etc"
        }
      ],
      "essential": true
    },
```

Erstellen Sie einen anderen Container, der mit `volumesFrom` alle Volumes mountet, die dem Container `web` zugeordnet sind. Alle Volumes im `web`-Container werden ebenfalls im `busybox`-Container gemountet. Das schließt das in der Dockerfile angegebene Volume ein, das zum Erstellen des `my-repo/httpd_dockerfile_volume`-Images verwendet wurde.

```
    {
      "name": "busybox",
      "image": "busybox",
      "volumesFrom": [
        {
          "sourceContainer": "web"
        }
      ],
      "cpu": 100,
      "memory": 500,
      "entryPoint": [
        "sh",
        "-c"
      ],
      "command": [
        "echo $(date) > /usr/local/apache2/htdocs/empty_volume/date && echo $(date)
 > /usr/local/apache2/htdocs/host_etc/date && echo $(date) > /usr/local/apache2/
htdocs/dockerfile_volume/date"
      ],
      "essential": false
    }
  ]
}
```

Wenn diese Aufgabe ausgeführt wird, mounten die beiden Container die Volumes, und der `command` im `busybox`-Container schreibt das Datum und die Uhrzeit in eine Datei. Diese Datei

heißt `date` in jedem der Volume-Ordner. Die Ordner sind dann auf der Website sichtbar, die durch den Container `web` angezeigt wird.

> **Note**
>
> Da der `busybox`-Container einen Kurzbefehl ausführt und dann beendet wird, muss er in der Containerdefinition auf `"essential": false` gesetzt werden. Andernfalls wird die gesamte Aufgabe gestoppt, wenn er beendet wird.

Verwaltung des Container-Swap-Speicherplatzes auf Amazon ECS

Mit Amazon ECS können Sie die Nutzung des Auslagerungsspeicherplatzes auf Ihren Linux-basierten Amazon-EC2-Instances auf Containerebene steuern. Bei der Verwendung einer Auslagerungskonfiguration pro Container kann die Auslagerung für jeden Container innerhalb einer Aufgabendefinition aktiviert oder deaktiviert sein. Für diejenigen, für die sie aktiviert ist, kann die maximale Menge des verwendeten Auslagerungsbereichs begrenzt sein. Beispielsweise kann die Auslagerung bei latenzkritischen Containern deaktiviert sein. Im Gegensatz dazu kann bei Containern mit hohem transienten Speicherbedarf Swap aktiviert sein, um die Wahrscheinlichkeit von out-of-memory Fehlern zu verringern, wenn der Container unter Last ist.

Die Auslagerungskonfiguration für einen Container wird mit den folgenden Container-Definitionsparametern verwaltet.

`maxSwap`

Die Gesamtmenge des Auslagerungsspeichers (in MiB), den ein Container verwenden kann. Dieser Parameter ist in die Option `--memory-swap` zur Docker-Ausführung übersetzt, wobei der Wert die Summe aus dem Container-Speicher und dem Wert `maxSwap` ist.

Wenn als `maxSwap`-Wert `0` angegeben wird, verwendet der Container keine Auslagerung. Zulässige Werte sind `0` oder eine beliebige positive Ganzzahl. Wenn der Parameter `maxSwap` weggelassen wird, verwendet der Container die Swap-Konfiguration für die Container-Instance, auf der er ausgeführt wird. Es muss ein Wert für `maxSwap` festgelegt werden, damit der Parameter `swappiness` verwendet werden kann.

`swappiness`

Auf diese Weise können Sie das Speicherauslagerungsverhalten eines Containers optimieren. Ein `swappiness`-Wert von `0` führt dazu, dass keine Auslagerung stattfindet, es sei denn,

dies ist absolut notwendig. Ein `swappiness`-Wert von `100` führt dazu, dass Seiten aggressiv ausgelagert werden. Akzeptierte Werte sind Ganzzahlen zwischen `0` und `100`. Wenn der Parameter `swappiness` nicht angegeben wird, wird der Standardwert `60` verwendet. Wenn kein Wert für `maxSwap` angegeben ist, wird dieser Parameter ignoriert. Dieser Parameter wird der Option `--memory-swappiness` für die Docker-Ausführung zugeordnet.

Im folgenden Beispiel wird die JSON-Syntax bereitgestellt.

```
"containerDefinitions": [{
        ...
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Überlegungen

Beachten Sie Folgendes, wenn Sie eine Auslagerungskonfiguration pro Container verwenden.

- Der Auslagerungsbereich muss auf der Amazon-EC2-Instance, die Ihre Aufgaben hostet, aktiviert und zugewiesen werden, damit die Container verwendet werden können. Für die von Amazon ECS optimierten AMIs ist Auslagern nicht standardmäßig aktiviert. Sie müssen die Auslagerung auf der Instance aktivieren, um dieses Feature verwenden zu können. Weitere Informationen finden Sie unter Instance Store Swap Volumes im Amazon EC2 EC2-Benutzerhandbuch oder Wie weise ich mithilfe einer Swap-Datei Speicher zu, der als Auslagerungsspeicher in einer Amazon EC2 EC2-Instance fungiert? .
- Die Container-Definitionsparameter für den Auslagerungsbereich werden nur für Aufgabendefinitionen unterstützt, die den Starttyp EC2 angeben. Diese Parameter werden nicht für Aufgabendefinitionen unterstützt, die nur für die Verwendung von Amazon ECS auf Fargate bestimmt sind.
- Dieses Feature wird nur für Linux-Container unterstützt. Windows-Container werden derzeit nicht unterstützt.
- Wenn die Container-Definitionsparameter `maxSwap` und `swappiness` in einer Aufgabendefinition weggelassen werden, hat jeder Container den `swappiness`-Standardwert `60`. Darüber hinaus ist die gesamte Swap-Nutzung auf das Zweifache des Speichers des Containers begrenzt.

- Wenn Sie Aufgaben auf Amazon Linux 2023 verwenden, wird der `swappiness`-Parameter nicht unterstützt.

Unterschiede bei der Amazon ECS-Aufgabendefinition für den Fargate-Starttyp

Um Fargate verwenden zu können, müssen Sie Ihre Aufgabendefinition so konfigurieren, dass sie den Fargate-Starttyp verwendet. Bei der Verwendung von Fargate sind weitere Überlegungen zu beachten.

Aufgabendefinitionsparameter

Aufgaben, die den Fargate-Starttyp verwenden, unterstützen nicht alle verfügbaren Amazon-ECS-Aufgabendefinitionsparameter. Einige Parameter werden generell nicht unterstützt, bei anderen weicht deren Verhalten für Fargate-Aufgaben ab.

Die folgenden Aufgabendefinitionsparameter sind in Fargate-Aufgaben nicht gültig:

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `extraHosts`
- `gpu`
- `ipcMode`
- `links`
- `placementConstraints`
- `privileged`
- `maxSwap`
- `swappiness`

Die folgenden Aufgabendefinitionsparameter sind in Fargate-Aufgaben gültig, weisen jedoch Einschränkungen auf, die zu beachten sind:

- `linuxParameters` – Bei der Angabe von Linux-spezifischen Optionen, die auf den Container angewendet werden, ist `CAP_SYS_PTRACE` die einzige Kapazität für `capabilities`, die Sie hinzufügen können. Die Parameter `devices`, `sharedMemorySize` und `tmpfs` werden nicht unterstützt. Weitere Informationen finden Sie unter Linux-Parameter.
- `volumes` – Fargate-Aufgaben unterstützen nur Bind-Mount-Host-Volumes, der `dockerVolumeConfiguration`-Parameter wird daher nicht unterstützt. Weitere Informationen finden Sie unter Datenträger.
- `cpu` – Für Windows-Container auf AWS Fargate kann der Wert nicht kleiner als 1 vCPU sein.

Um sicherzustellen, dass die Aufgabendefinition für Fargate validiert wird, können Sie beim Registrieren der Aufgabendefinition Folgendes angeben:

- Geben Sie AWS Management Console im Feld Erfordert Kompatibilitäten an. `FARGATE`
- Geben Sie im AWS CLI die `--requires-compatibilities` Option an.
- Legen Sie in der Amazon ECS API das Flag `requiresCompatibilities` fest.

Betriebssysteme und Architekturen

Wenn Sie eine Aufgaben- und Containerdefinition für AWS Fargate konfigurieren, müssen Sie das Betriebssystem angeben, das der Container ausführt. Die folgenden Betriebssysteme werden für AWS Fargate unterstützt:

- Amazon Linux 2

> **Note**
>
> Linux-Container verwenden nur den Kernel und die Kernelkonfiguration des Host-Betriebssystems. Die Kernel-Konfiguration umfasst beispielsweise die `sysctl`-Systemsteuerelemente. Ein Linux-Container-Image kann aus einem Basis-Image erstellt werden, das die Dateien und Programme aus jeder Linux-Verteilung enthält. Wenn die CPU-Architektur übereinstimmt, können Sie Container von jedem Linux-Container-Image auf jedem Betriebssystem aus ausführen.

- Windows Server 2019 Voll
- Windows Server 2019 Kern
- Windows Server 2022 Voll

- Windows Server 2022 Kern

Wenn Sie Windows-Container auf AWS Fargate ausführen, benötigen Sie eine X86_64-CPU-Architektur.

Wenn Sie Linux-Container auf AWS Fargate ausführen, können Sie die X86_64-CPU-Architektur oder die ARM64-Architektur für Ihre ARM-basierten Anwendungen verwenden. Weitere Informationen finden Sie unter the section called “Aufgabendefinitionen für 64-Bit-ARM-Workloads”.

CPU und Arbeitsspeicher der Aufgabe

Amazon-ECS-Aufgabendefinitionen für AWS Fargate erfordern, dass Sie CPU und Speicher auf Aufgabenebene angeben. Sie können CPU- und Speicherauslastung zwar für Fargate-Aufgaben auch auf Container-Ebene festlegen, dies ist jedoch optional. In den meisten Fällen reicht es aus, diese Ressourcen nur auf Aufgabenebene festzulegen. Die folgende Tabelle zeigt die gültigen Kombinationen von CPU- und Speicherauslastung auf Aufgabenebene. Sie können Speicherwerte in der Aufgabendefinition als Zeichenfolge in MiB oder GB angeben. Sie können beispielsweise einen Speicherwert entweder `3072` in MiB oder `3 GB` in GB angeben. Sie können CPU-Werte in der JSON-Datei als Zeichenfolge in CPU-Einheiten oder virtuellen CPUs (vCPUs) angeben. Sie können beispielsweise einen CPU-Wert entweder in CPU-Einheiten oder `1024` `1 vCPU` in vCPUs angeben.

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Zwischen 4 GB und 16 GB in 1-GB-Schritten	Linux, Windows
4096 (4 vCPU)	Zwischen 8 GB und 30 GB in 1-GB-Schritten	Linux, Windows
8 192 (8 vCPU)	Zwischen 16 GB und 60 GB in 4-GB-Schritten	Linux

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
Note Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.		
16 384 (16 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 32 GB und 120 GB in 8-GB-Schritten	Linux

Aufgabenvernetzung

Amazon-ECS-Aufgaben für AWS Fargate setzen den `awsvpc`-Netzwerkmodus voraus, der für jede Aufgabe eine Elastic-Network-Schnittstelle bereitstellt. Wenn Sie mit diesem Netzwerkmodus eine Aufgabe ausführen oder einen Service erstellen, müssen Sie mindestens ein Subnetz und mindestens eine Sicherheitsgruppe für die Netzwerkschnittstelle festlegen.

Wenn Sie keine öffentlichen Subnetze verwenden, legen Sie fest, ob Sie eine öffentliche IP-Adresse für die Netzwerkschnittstelle bereitstellen möchten. Damit die Fargate-Aufgabe in einem öffentlichen Subnetz Container-Images abrufen kann, muss der Elastic-Network-Schnittstelle der Aufgabe eine öffentliche IP-Adresse sowie eine Route zum Internet oder einem NAT-Gateway zugewiesen werden, um Anfragen an das Internet weiterzuleiten. Damit eine Fargate-Aufgabe in einem privaten Subnetz Container-Images abrufen kann, benötigen Sie ein NAT-Gateway im Subnetz zur Weiterleitung von Anforderungen an das Internet. Wenn Sie Ihre Container-Images in Amazon ECR hosten, können Sie Amazon ECR so konfigurieren, dass ein Interface-VPC-Endpunkt verwendet wird. In diesem Fall wird die private IPv4-Adresse der Aufgabe für den Image-Pull verwendet. Weitere Informationen zu Amazon ECR-Schnittstellen-Endpunkten finden Sie unter Amazon ECR-Schnittstellen-VPC Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für den `networkConfiguration` Abschnitt für einen Fargate-Dienst:

```
"networkConfiguration": {
   "awsvpcConfiguration": {
      "assignPublicIp": "ENABLED",
      "securityGroups": [ "sg-12345678" ],
      "subnets": [ "subnet-12345678" ]
   }
}
```

Aufgabenressourcenlimits

Amazon-ECS-Aufgabendefinitionen für Linux-Container auf AWS Fargate unterstützen den Parameter `ulimits` zum Definieren der Ressourcenlimits, die für einen Container festgelegt werden sollen.

Amazon-ECS-Aufgabendefinitionen für Windows auf AWS Fargate unterstützen den Parameter `ulimits` zum Definieren der Ressourcenlimits, die für einen Container eingestellt werden sollen, nicht.

Amazon-ECS-Aufgaben, die auf Fargate gehosted werden, verwenden die Standardwerte für Ressourcenlimits, die vom Betriebssystem gesetzt wurden. Ausgenommen ist der Ressourcenlimitparameter `nofile`. Das Ressourcenlimit `nofile` beschränkt die Anzahl der geöffneten Dateien, die ein Container verwenden kann. Auf Fargate ist die `nofile`-Standardeinstellung für die weiche Grenze `1024` und die harte Grenze `65535`. Sie können die Werte beider Grenzwerte auf bis zu `1048576` festlegen.

Das folgende ist ein Beispiel-Snippet für Aufgabendefinitionen, das zeigt, wie eine benutzerdefinierte `nofile`-Grenze, die verdoppelt wurde:

```
"ulimits": [
    {
       "name": "nofile",
       "softLimit": 2048,
       "hardLimit": 8192
    }
]
```

Weitere Informationen zu den anderen Ressourcenlimits, die angepasst werden können, finden Sie unter Ressourcenlimits.

Protokollierung

Protokollieren von Ereignissen

Amazon ECS protokolliert die Aktionen, die es ausführt EventBridge. Sie können Amazon ECS-Ereignisse verwenden EventBridge , um nahezu in Echtzeit Benachrichtigungen über den aktuellen Status Ihrer Amazon ECS-Cluster, -Services und -Tasks zu erhalten. Darüber hinaus können Sie Aktionen zur Reaktion auf diese Ereignisse automatisieren. Weitere Informationen finden Sie unter Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge.

Aufgabenlebenszyklus protokollieren

Aufgaben, die auf Fargate ausgeführt werden, veröffentlichen Zeitstempel, um die Aufgabe über die Status des Aufgabenlebenszyklus hinweg zu verfolgen. Sie können die Zeitstempel in den Aufgabendetails in den AWS Management Console und anhand der Beschreibung der Aufgabe in den SDKs AWS CLI und einsehen. Mithilfe der Zeitstempel können Sie beispielsweise auswerten, wie viel Zeit die Aufgabe für das Herunterladen der Container-Images aufgewendet hat, und entscheiden, ob Sie die Größe des Container-Images optimieren oder Seekable-OCI-Indizes verwenden sollten. Weitere Informationen über Container-Image-Verfahren finden Sie unter Bewährte Methoden für Amazon ECS-Container-Images.

Anwendungsprotokollierung

Amazon-ECS-Aufgabendefinitionen für AWS Fargate unterstützen die Protokolltreiber `awslogs`, `splunk` und `awsfirelens` für die Protokollkonfiguration.

Der `awslogs` Protokolltreiber konfiguriert Ihre Fargate-Aufgaben so, dass Protokollinformationen an Amazon CloudWatch Logs gesendet werden. Das folgende Beispiel zeigt einen Ausschnitt einer Aufgabendefinition, in der der Protokolltreiber `awslogs` konfiguriert wurde:

```
"logConfiguration": {
   "logDriver": "awslogs",
   "options": {
      "awslogs-group" : "/ecs/fargate-task-definition",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
   }
}
```

Weitere Informationen zur Verwendung des `awslogs` Log-Treibers in einer Aufgabendefinition zum Senden Ihrer Container-Logs an CloudWatch Logs finden Sie unter. Amazon ECS-Protokolle senden an CloudWatch

Weitere Informationen zum `awsfirelens`-Protokolltreiber in einer Aufgabendefinition finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.

Weitere Informationen zur Verwendung des Protokolltreibers `splunk` in einer Aufgabendefinition finden Sie unter `splunk`Protokolltreiber.

Aufgabenspeicher

Für Amazon-ECS-Aufgaben, die auf Fargate gehostet werden, werden die folgenden Speichertypen unterstützt:

- Amazon EBS-Volumes bieten kostengünstigen, dauerhaften und leistungsstarken Blockspeicher für datenintensive containerisierte Workloads. Weitere Informationen finden Sie unter Verwenden Sie Amazon EBS-Volumes mit Amazon ECS.
- Amazon EFS-Volumes für persistenten Speicher. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.
- Binden Sie Halterungen für den flüchtigen Speicher. Weitere Informationen finden Sie unter Bind-Mounts mit Amazon ECS verwenden.

Lazy Loading von Container-Images mit Seekable OCI (SOCI)

Amazon-ECS-Aufgaben auf Fargate, welche die Linux-Plattformversion `1.4.0` verwenden, können Seekable OCI (SOCI) einsetzen, um Aufgaben schneller zu starten. Mit SOCI verbringen Container nur wenige Sekunden mit dem Abruf von Images, bevor sie starten können. So bleibt Zeit für die Einrichtung der Umgebung und die Instanziierung der Anwendung, während das Image im Hintergrund heruntergeladen wird. Dies wird als Lazy Loading bezeichnet. Wenn Fargate eine Amazon-ECS-Aufgabe startet, erkennt Fargate automatisch, ob ein SOCI-Index für ein Image in der Aufgabe vorhanden ist, und startet den Container, ohne darauf zu warten, dass das gesamte Image heruntergeladen wird.

Bei Containern, die ohne SOCI-Indizes ausgeführt werden, werden Container-Images vollständig heruntergeladen, bevor der Container gestartet wird. Dieses Verhalten ist auf allen anderen Plattformversionen von Fargate und auf dem Amazon-ECS-optimierten AMI auf Amazon-EC2 Instances gleich.

Seekable OCI Indexes

Seekable OCI (SOCI) ist eine von entwickelte Open-Source-Technologie, mit der Container schneller gestartet werden können AWS , indem das Container-Image langsam geladen wird. SOCI erstellt einen Index (SOCI-Index) der Dateien in einem vorhandenen Container-Image. Dieser Index hilft dabei, Container schneller zu starten, indem er die Möglichkeit bietet, eine einzelne Datei aus einem Container-Image zu extrahieren, bevor das gesamte Image heruntergeladen wird. Der SOCI-Index muss als Artefakt im selben Repository wie das Image in der Container-Registrierung gespeichert werden. Sie sollten nur SOCI-Indizes aus vertrauenswürdigen Quellen verwenden, da der Index die autorisierende Quelle für den Inhalt des Images ist. Weitere Informationen finden Sie unter Einführen von Seekable OCI für Lazy Loading von Container-Images.

Überlegungen

Wenn Sie möchten, dass Fargate einen SOCI-Index verwendet, um Container-Images in einer Aufgabe verzögert zu laden, sollten Sie Folgendes berücksichtigen:

- Nur Aufgaben, die auf Linux-Plattformversion `1.4.0` ausgeführt werden, können SOCI-Indizes verwenden. Aufgaben, die Windows-Container auf Fargate ausführen, werden nicht unterstützt.
- Aufgaben, die auf X86_64- oder ARM64-CPU-Architektur ausgeführt werden, werden unterstützt. Linux-Aufgaben mit der ARM64-Architektur unterstützen den Fargate-Spot-Kapazitätsanbieter nicht.
- Container-Images in der Aufgabendefinition müssen SOCI-Indizes in derselben Container-Registrierung wie das Image aufbewahren.
- Container-Images in der Aufgabendefinition müssen in einer kompatiblen Image-Registry gespeichert werden. Im Folgenden sind die kompatiblen Registrys aufgeführt:
  - Private Amazon-ECR-Registrys.
- Es werden nur Container-Images unterstützt, die gzip-Komprimierung verwenden oder nicht komprimiert sind. Container-Images, die zstd-Komprimierung verwenden, werden nicht unterstützt.
- Wir empfehlen Ihnen, verzögertes Laden mit Container-Images zu versuchen, deren Größe größer als 250 MiB komprimiert ist. Es ist weniger wahrscheinlich, dass die Zeit zum Laden kleinerer Image verkürzt werden kann.
- Da Lazy Loading ändern kann, wie lange es dauert, bis Ihre Aufgaben gestartet werden, müssen Sie möglicherweise verschiedene Timeouts ändern, z. B. den Kulanzzeitraum für die Zustandsprüfung für Elastic Load Balancing.
- Wenn Sie verhindern möchten, dass ein Container-Image verzögert geladen wird, löschen Sie den SOCI-Index aus der Container-Registrierung. Wenn ein Container-Image in der Aufgabe

eine der Voraussetzungen nicht erfüllt, wird dieses Container-Image mit der Standardmethode heruntergeladen.

Erstellen eines Seekable-OCI-Indexes

Damit ein Container-Image verzögert geladen werden kann, benötigt es einen SOCI-Index (eine Metadatendatei), der erstellt und zusammen mit dem Container-Image im Container-Image-Repository gespeichert wird. Um einen SOCI-Index zu erstellen und zu pushen, können Sie das Open-Source-CLI-Tool soci-snapshotter verwenden. GitHub Oder Sie können den SOCI Index Builder bereitstellen. CloudFormation AWS Dies ist eine serverlose Lösung, die automatisch einen SOCI-Index erstellt und überträgt, wenn ein Container-Image an Amazon ECR übertragen wird. Weitere Informationen zur Lösung und den Installationsschritten finden Sie unter CloudFormation AWS SOCI Index Builder unter. GitHub Mit dem CloudFormation AWS SOCI Index Builder können Sie die ersten Schritte mit SOCI automatisieren, während das Open-Source-Tool SOCI mehr Flexibilität bei der Indexgenerierung bietet und die Indexgenerierung in Ihre CI/CD-Pipelines (Continuous Integration and Continuous Delivery) integrieren kann.

> **Note**
>
> Damit der SOCI-Index für ein Image erstellt werden kann, muss das Image im containerd-Imagespeicher des `soci-snapshotter` ausführenden Computers vorhanden sein. Wenn sich das Image im Docker-Bildspeicher befindet, kann das Bild Image gefunden werden.

Überprüfen, ob eine Aufgabe Lazy Loading verwendet

Um zu überprüfen, ob eine Aufgabe mit Hilfe von SOCI verzögert geladen wurde, überprüfen Sie den Metadaten-Endpunkt der Aufgabe von innerhalb der Aufgabe. Wenn Sie den Aufgabenmetadaten-Endpunkt Version 4 abfragen, gibt es ein `Snapshotter`-Feld im Standardpfad für den Container, von dem aus Sie die Abfrage durchführen. Darüber hinaus gibt es `Snapshotter`-Felder für jeden Container im `/task`-Pfad. Der Standardwert für dieses Feld ist`overlayfs`, und dieses Feld ist auf gesetzt, wenn SOCI verwendet wird. `soci`

Unterschiede bei der Amazon ECS-Aufgabendefinition für EC2-Instances, auf denen Windows ausgeführt wird

Aufgaben, die auf EC2-Windows-Instances ausgeführt werden, unterstützen nicht alle verfügbaren Amazon ECS-Aufgabendefinitionsparameter. Einige Parameter werden überhaupt nicht unterstützt, und andere verhalten sich anders.

Die folgenden Aufgabendefinitionsparameter werden für Amazon EC2 EC2-Windows-Aufgabendefinitionen nicht unterstützt:

- `containerDefinitions`
  - `disableNetworking`
  - `dnsServers`
  - `dnsSearchDomains`
  - `extraHosts`
  - `links`
  - `linuxParameters`
  - `privileged`
  - `readonlyRootFilesystem`
  - `user`
  - `ulimits`
- `volumes`
  - `dockerVolumeConfiguration`
- `cpu`

  Es wird empfohlen, für Windows-Container eine CPU auf Container-Ebene festzulegen.
- `memory`

  Es wird empfohlen, für Windows-Container einen Arbeitsspeicher auf Container-Ebene festzulegen.
- `proxyConfiguration`
- `ipcMode`
- `pidMode`
- `taskRoleArn`

Die Funktionen der IAM-Rollen für Aufgaben auf EC2-Windows-Instances erfordern eine zusätzliche Konfiguration, aber ein Großteil dieser Konfiguration ähnelt der Konfiguration von IAM-Rollen für Aufgaben auf Linux-Container-Instances. Weitere Informationen finden Sie unter the section called “ Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance”.

Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole

Sie können eine Aufgabendefinition mithilfe der Konsole oder durch Bearbeiten einer JSON-Datei erstellen.

JSON-Validierung

Der JSON-Editor der Amazon-ECS-Konsole validiert Folgendes in der JSON-Datei:

- Die Datei ist eine gültige JSON-Datei.
- Die Datei enthält keine überflüssigen Schlüssel.
- Die Datei enthält den `familyName` Parameter.
- Es gibt mindestens einen Eintrag unter`containerDefinitions`.

AWS CloudFormation stapelt

Das folgende Verhalten gilt für Aufgabendefinitionen, die vor dem 12. Januar 2023 in der neuen Amazon ECS-Konsole erstellt wurden.

Wenn Sie eine Aufgabendefinition erstellen, erstellt die Amazon ECS-Konsole automatisch einen CloudFormation Stack, dessen Name mit beginnt`ECS-Console-V2-TaskDefinition-`. Wenn Sie das AWS CLI oder ein AWS SDK verwendet haben, um die Registrierung der Aufgabendefinition aufzuheben, müssen Sie den Aufgabendefinitionsstapel manuell löschen. Weitere Informationen finden Sie im AWS CloudFormation Benutzerhandbuch unter Löschen eines Stacks.

Für Aufgabendefinitionen, die nach dem 12. Januar 2023 erstellt wurden, wird kein CloudFormation Stapel automatisch für sie erstellt.

Verfahren

Amazon ECS console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie im Menü Neue Aufgabendefinition erstellen die Option Neue Aufgabendefinition erstellen aus.
4. Geben Sie für Task definition family (Aufgabendefinitions-Familie) einen eindeutigen Namen für die Aufgabendefinition an.
5. Wählen Sie als Starttyp die Anwendungsumgebung aus. Die Standardeinstellung für die Konsole ist AWS Fargate(serverlos). Amazon ECS verwendet diesen Wert, um eine Validierung durchzuführen, um sicherzustellen, dass die Aufgabendefinitionsparameter für den Infrastrukturtyp gültig sind.
6. Wählen Sie für Operating system/architecture (Betriebssystem/Architektur) das Betriebssystem und die CPU-Architektur für die Aufgabe aus.

   Um Ihre Aufgabe auf einer 64-Bit-ARM-Architektur auszuführen, wählen Sie Linux/ARM64. Weitere Informationen finden Sie unter the section called “Laufzeit-Plattform”.

   Um Ihre AWS Fargate-Aufgaben auf Windows-Containern auszuführen, wählen Sie ein unterstütztes Windows-Betriebssystem aus. Weitere Informationen finden Sie unter the section called “Betriebssysteme und Architekturen”.
7. Wählen Sie für Task size (Aufgabengröße) die CPU- und Arbeitsspeicherwerte aus, die für die Aufgabe reserviert werden sollen. Der CPU-Wert wird als vCPUs angegeben und der Arbeitsspeicher als GB angegeben.

   Für Aufgaben, die auf Fargate gehostet werden, zeigt die folgende Tabelle die gültigen CPU- und Arbeitsspeicher-Kombinationen.

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Zwischen 4 GB und 16 GB in 1-GB-Schritten	Linux, Windows
4096 (4 vCPU)	Zwischen 8 GB und 30 GB in 1-GB-Schritten	Linux, Windows
8 192 (8 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 16 GB und 60 GB in 4-GB-Schritten	Linux
16 384 (16 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 32 GB und 120 GB in 8-GB-Schritten	Linux

Für auf Amazon EC2 gehostete Aufgaben liegen die unterstützten Aufgaben-CPU-Werte zwischen 128 CPU-Einheiten (0,125 vCPUs) und 10240 CPU-Einheiten (10 vCPUs). Um den Speicherwert in GB anzugeben, geben Sie nach dem Wert GB ein. Um beispielsweise den Speicherwert auf 3 GB festzulegen, geben Sie 3 GB ein.

> **Note**
>
> CPU- und Speicherparameter auf Aufgabenebene werden für Windows-Container ignoriert.

8. Wählen Sie für Network mode (Netzwerkmodus) den zu verwendenden Netzwerkmodus aus. Der Standardmodus ist awsvpc. Weitere Informationen finden Sie unter Amazon-ECS-Aufgabenvernetzung.

   Wenn Sie Bridge wählen, geben Sie unter Port-Zuordnungen für Host-Port die Portnummer der Container-Instance ein, die Sie für Ihren Container reservieren möchten.

9. (Optional) Erweitern Sie den Abschnitt Aufgabenrollen, um die AWS Identity and Access Management (IAM-) Rollen für die Aufgabe zu konfigurieren:

   a. Wählen Sie für Task role (Aufgabenrolle) die IAM-Rolle aus, die Sie der Aufgabe zuweisen möchten. Eine Task-IAM-Rolle gewährt den Containern in einer Aufgabe die Berechtigungen, AWS API-Operationen aufzurufen.

   b. Wählen Sie für die Aufgabenausführungsrolle die Rolle aus.

      Informationen darüber, wann Sie eine Aufgabenausführungsrolle verwenden sollten, finden Sie unter the section called “IAM-Aufgabenausführungsrolle”. Wenn Sie die Rolle nicht benötigen, wählen Sie Keine.

10. Führen Sie für jeden in Ihrer Aufgabendefinition zu definierenden Container die folgenden Schritte aus.

    a. Geben Sie unter Name einen Namen für den Container ein.

    b. Geben Sie für Image URI (Image-URI) das Image an, das zum Starten eines Containers verwendet werden soll. Bilder in der Amazon ECR Public Gallery-Registrierung können nur mit dem Namen der Amazon ECR Public Registry angegeben werden. Wenn beispielsweise angegeben `public.ecr.aws/ecs/amazon-ecs-agent:latest` ist, wird der Amazon Linux-Container verwendet, der in der Amazon ECR Public Gallery gehostet wird. Geben Sie für alle anderen Repositorys das Repository an, indem Sie entweder das `repository-url/image:tag` oder `repository-url/image@digest` -Format verwenden.

    c. Wenn sich Ihr Image in einer privaten Registrierung außerhalb von Amazon ECR befindet, aktivieren Sie unter Private Registrierung die Authentifizierung mit privater

Registrierung. Geben Sie dann unter Secrets-Manager-ARN oder -Name den Amazon-Ressourcennamen (ARN) des Secrets ein.

d. Wenn in Ihrer Aufgabendefinition zwei oder mehr Container definiert sind, können Sie für Essential-Container angeben, ob der Container als essenziell betrachtet werden soll. Wenn ein Container als Essential markiert ist und dieser Container beendet wird, wird die Aufgabe gestoppt. Jede Aufgabendefinition muss mindestens einen essenziellen Container enthalten.

e. Ein Port-Mapping erlaubt es dem Container, auf Ports auf dem Host zuzugreifen, um Datenverkehr zu senden oder zu empfangen. Führen Sie unter Port-Mappings einen der folgenden Schritte aus:

   - Wenn Sie den awsvpc-Netzwerkmodus verwenden, geben Sie für Container port (Container-Port) und Protocol (Protokoll) das Port-Mapping an, das für den Container verwendet werden soll.
   - Wenn Sie den bridge-Netzwerkmodus verwenden, wählen Sie für Container port (Container-Port) und Protocol (Protokoll) die Port-Zuweisung, die für den Container verwendet werden soll.

   Wählen Sie Add more port mappings (Weitere Port-Mappings hinzufügen) aus, um zusätzliche Container-Port-Mappings anzugeben.

f. Wenn Sie dem Container schreibgeschützten Zugriff auf das Root-Dateisystem gewähren möchten, wählen Sie für Schreibgeschütztes Root-Dateisystem die Option Schreibgeschützter Zugriff.

g. (Optional) Gehen Sie unter Grenzwerte für die Ressourcenzuweisung wie folgt vor, um die CPU-, GPU- und Speichergrenzwerte auf Containerebene zu definieren, die sich von den Werten auf Aufgabenebene unterscheiden:

   - Geben Sie für CPU die Anzahl der CPU-Einheiten ein, die der Amazon ECS-Container-Agent für den Container reserviert.
   - Geben Sie für GPU die Anzahl der GPU-Einheiten für die Container-Instance ein.

     Eine Amazon-EC2-Instance mit GPU-Unterstützung hat 1 GPU-Einheit für jede GPU. Weitere Informationen finden Sie unter the section called “Aufgabendefinitionen für GPU-Workloads”.

- Geben Sie unter Festes Speicherlimit die Speichermenge in GB ein, die dem Container zur Verfügung gestellt werden soll. Wenn der Container versucht, die harte Grenze zu überschreiten, stoppt der Container.
- Der Daemon Docker 20.10.0 oder höher reserviert mindestens 6 Mebibyte (MiB) Speicher für einen Container. Geben Sie also nicht weniger als 6 MiB Speicher für Ihre Container an.

  Der Daemon Docker 19.03.13-ce oder früher reserviert mindestens 4 MiB Speicher für einen Container, geben Sie also nicht weniger als 4 MiB Speicher für Ihre Container an.
- Geben Sie unter Memory Soft Limit die weiche Grenze (in GB) des Speichers ein, der für den Container reserviert werden soll.

  Wenn der Systemspeicher strittig ist, versucht Docker den Arbeitsspeicher des Containers innerhalb der weichen Grenze zu halten. Wenn Sie keinen Speicher auf Aufgabenebene angeben, müssen Sie eine Ganzzahl ungleich Null für einen oder beide Werte von Memory Hard Limit und Memory Soft Limit angeben. Wenn Sie beide angeben, muss Memory Hard Limit größer sein als Memory Soft Limit.

  Diese Funktion wird in Windows-Containern nicht unterstützt.

h. (Optional) Erweitern Sie den Abschnitt Umgebungsvariablen, um Umgebungsvariablen anzugeben, die in den Container eingefügt werden sollen. Sie können Umgebungsvariablen entweder einzeln mithilfe von Schlüssel-Wert-Paaren oder in großen Mengen angeben, indem Sie eine Umgebungsvariablendatei angeben, die in einem Amazon S3 S3-Bucket gehostet wird. Informationen zum Formatieren einer Umgebungsvariablendatei finden Sie unter. Übergeben Sie eine einzelne Umgebungsvariable an einen Amazon ECS-Container

   Wenn Sie eine Umgebungsvariable für den geheimen Speicher angeben, geben Sie unter Schlüssel den geheimen Namen ein. Geben Sie ValueFromdann für den vollständigen ARN des Systems Manager Parameter Store Secret oder Secrets Manager Secret ein

i. (Optional) Wählen Sie die Option Use log collection (Protokollerfassung verwenden), um eine Protokollkonfiguration anzugeben. Für jeden verfügbaren Protokolltreiber gibt es Protokolltreiberoptionen, die angegeben werden müssen. Die Standardoption sendet Container-Logs an Amazon CloudWatch Logs. Die anderen Optionen für den Protokolltreiber werden mithilfe von konfiguriert AWS FireLens. Weitere Informationen

finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.

Im Folgenden wird jedes Container-Protokollziel ausführlicher beschrieben.

- Amazon CloudWatch — Konfigurieren Sie die Aufgabe, um Container-Logs an CloudWatch Logs zu senden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, mit denen in Ihrem Namen eine CloudWatch Protokollgruppe erstellt wird. Um einen anderen Protokollgruppen-Namen anzugeben, ändern Sie die Werte der Treiberoption.
- Protokolle exportieren nach Splunk — Konfigurieren Sie die Aufgabe so, dass Containerprotokolle an den Splunk Treiber gesendet werden, der die Protokolle an einen Remotedienst sendet. Sie müssen die URL zu Ihrem Splunk Webservice eingeben. Das Splunk Token ist als geheime Option angegeben, da es als vertrauliche Daten behandelt werden kann.
- Protokolle nach Amazon Data Firehose exportieren — Konfigurieren Sie die Aufgabe so, dass Container-Protokolle an Firehose gesendet werden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, die das Protokoll an einen Firehose-Lieferstream senden. Um einen anderen Namen für den Bereitstellungsdatenstrom anzugeben, ändern Sie die Werte der Treiberoption.
- Protokolle nach Amazon Kinesis Data Streams exportieren — Konfigurieren Sie die Aufgabe so, dass Container-Protokolle an Kinesis Data Streams gesendet werden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, mit denen Protokolle an einen Kinesis Data Streams Streams-Stream gesendet werden. Um einen anderen Datenstrom-Namen anzugeben, ändern Sie die Werte der Treiberoption.
- Protokolle nach Amazon OpenSearch Service exportieren — Konfigurieren Sie die Aufgabe so, dass Container-Protokolle an eine OpenSearch Service-Domain gesendet werden. Die Optionen für den Protokolltreiber müssen bereitgestellt werden.
- Protokolle nach Amazon S3 exportieren — Konfigurieren Sie die Aufgabe so, dass Container-Protokolle an einen Amazon S3 S3-Bucket gesendet werden. Die Standardoptionen für den Protokolltreiber sind verfügbar, Sie müssen jedoch einen gültigen Amazon S3 S3-Bucket-Namen angeben.

j. (Optional) Konfigurieren Sie zusätzliche Container-Parameter.

So konfigurieren Sie diese Option	Vorgehensweise	
Healthcheck Mit diesen Befehlen wird festgestellt, ob ein Container fehlerfrei ist. Weitere Informationen finden Sie unter Ermitteln Sie den Zustand von Amazon ECS-Aufgaben mithilfe von Container-Zustandsprüfungen.	Erweitern Sie HealthCheckund konfigurieren Sie dann die folgenden Elemente: • Geben Sie unter Command (Befehl) eine durch Kommas getrennte Liste von Befehlen ein. Sie können die Befehle mit `CMD` starten, um die Befehlsargumente direkt auszuführen, oder mit `CMD-SHELL` , um den Befehl mit der Standard-Shell des Containers auszuführen. Ist nichts davon angegeben, wird `CMD` verwendet. • Geben Sie als Interval (Intervall) die Anzahl der Sekunden zwischen den einzelnen Zustandsprüfungen ein. Die gültigen Werte liegen zwischen 5 und 30. • Geben Sie für Timeout die Zeitspanne (in Sekunden) ein, die	

So konfigurieren Sie diese Option	Vorgehensweise	
	abgewartet werden soll, bis eine Zustandsprüfung erfolgreich war, bevor diese als Fehlschlag gewertet wird. Die gültigen Werte liegen zwischen 2 und 60. • Geben Sie für Start period (Startzeitraum) die Zeitspanne (in Sekunden) ein, die abgewartet werden soll, bis ein Container gebootet ist, bevor die Befehle für die Zustandsprüfung ausgeführt werden. Die gültigen Werte liegen zwischen 0 und 300. • Geben Sie für Retries (Wiederholungen) ein, wie oft die Befehle zur Zustandsprüfung wiederholt werden sollen, wenn ein Fehler auftritt. Die gültigen Werte liegen zwischen 1 und 10.	

So konfigurieren Sie diese Option	Vorgehensweise	
Container-Timeouts Diese Optionen bestimmen, wann ein Container gestartet und gestoppt werden soll.	Erweitern Sie Container -Timeouts und konfiguri eren Sie dann Folgendes: • Um die Wartezeit zu konfigurieren, bis die Auflösung der Abhängigkeiten für einen Container aufgegeben wird, geben Sie unter Start-Tim eout die Anzahl der Sekunden ein. • Um die Wartezeit zu konfigurieren, bis der Container gestoppt wird, wenn er sich nicht normal von selbst beendet, geben Sie unter Stop-Timeout die Anzahl der Sekunden ein.	

So konfigurieren Sie diese Option	Vorgehensweise
Container-Netzwerk einstellungen Diese Optionen bestimmen, ob Netzwerke innerhalb eines Containers verwendet werden sollen.	Erweitern Sie Container-Netzwerkeinstellungen und konfigurieren Sie dann Folgendes: • Um Container-Netzwerke zu deaktivieren, wählen Sie Netzwerk ausschalten aus. • Um DNS-Server-IP-Adressen zu konfigurieren, die dem Container angezeigt werden, geben Sie unter DNS-Server die IP-Adressen der einzelnen Server in einer separaten Zeile ein. • Um DNS-Domänen für die Suche nach non-fully-qualified Hostnamen zu konfigurieren, die dem Container angezeigt werden, geben Sie im Feld DNS-Suchdomänen jede Domain in einer separaten Zeile ein. Das Muster ist **`^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$`** .

So konfigurieren Sie diese Option	Vorgehensweise	
	• Um den Container-Hostnamen zu konfigurieren, geben Sie im Feld Hostname den Container-Goat-Namen ein. • Um Hostnamen und IP-Adresszuordnungen hinzuzufügen, die an die `/etc/hosts` -Datei im Container angehängt werden, wählen Sie Zusätzlichen Host hinzufügen aus, und geben Sie dann für Hostname und IP-Adresse den Hostnamen und die IP-Adresse ein.	

So konfigurieren Sie diese Option	Vorgehensweise	
Docker-Konfiguration Diese überschreiben die Werte in derDockerfile.	Erweitern Sie die DockerKonfiguration und konfigurieren Sie dann die folgenden Elemente: • Geben Sie für Befehl einen ausführba ren Befehl für einen Container ein. Dieser Parameter entspricht `Cmd` dem Abschnitt Create a container der Docker Remote API und der `COMMAND` Option to`docker run` . Dieser Parameter überschreibt die `CMD` Anweisung in a. Dockerfile • Geben Sie als Einstiegs punkt den Docker ENTRYPOINT ein, der an den Container übergeben wird. Dieser Parameter ist dem `Entrypoint` Abschnitt Create a container der Docker Remote API und der `--`	

So konfigurieren Sie diese Option	Vorgehensweise	
	`entrypoint` Option to zugeordnet. `docker run` Dieser Parameter überschreibt die `ENTRYPOINT` Anweisung in a. Dockerfile • Geben Sie unter Arbeitsverzeichnis das Verzeichnis ein, in dem der Container alle bereitgestellten Einstiegspunkte und Befehlsanweisungen ausführen soll. Dieser Parameter entspricht `WorkingDir` dem Abschnitt Create a container der Docker Remote API und der `--workdir` Option to`docker run`. Dieser Parameter überschreibt die `WORKDIR` Anweisung in a. Dockerfile	

So konfigurieren Sie diese Option	Vorgehensweise
Ulimits Diese Werte überschreiben die Standardeinstellung für die Ressourcenkontingente des Betriebssystems. Dieser Parameter ordnet zu `Ulimits` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--ulimit` für die docker run zu.	Erweitern Sie Ressourcenlimits (ulimits) und wählen Sie dann Hinzufügen ulimit aus. Wählen Sie unter Limitname das Limit aus. Geben Sie dann für Soft Limit und Hard Limit die entsprechenden Werte ein. Um weitere hinzuzufügenulimits, wählen Sie Hinzufügen ulimit.
DockerBeschriftungen Diese Option fügt Ihrem Container Metadaten hinzu. Dieser Parameter ordnet zu `Labels` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--label` für die docker run zu.	Erweitern Sie DockerBeschriftungen, wählen Sie Schlüssel-Wert-Paar hinzufügen aus, und geben Sie dann Schlüssel und Wert ein. Um weitere Docker Labels hinzuzufügen, wählen Sie Schlüssel-Wert-Paar hinzufügen aus.

So konfigurieren Sie diese Option	Vorgehensweise	
Startup-Reihenfolge des Containers Diese Option definiert Abhängigkeiten für das Startup und Herunterf ahren von Containern. Ein Container kann mehrere Abhängigkeiten enthalten.	Erweitern Sie die Reihenfolge der Startup-Abhängigkeiten und konfigurieren Sie dann Folgendes: a. Wählen Sie Container abhängigkeit hinzufüge n aus. b. Wählen Sie für Container den Container aus. c. Wählen Sie unter Bedingung die Bedingung für die Startup-Abhängigkeit aus. Um eine zusätzliche Abhängigkeit hinzuzufü gen, wählen Sie Container abhängigkeit hinzufügen aus.	

k. (Optional) Wählen Sie Add more containers (Weiterer Container hinzufügen) aus, um der Aufgabendefinition zusätzliche Container hinzuzufügen.

11. (Optional) Der Bereich Speicher wird verwendet, um den Umfang des kurzlebigen Speichers für Aufgaben zu erweitern, die auf Fargate gehostet werden. Sie können diesen Abschnitt auch verwenden, um eine Datenvolumenkonfiguration für die Aufgabe hinzuzufügen.

- Geben Sie für Amount (Menge) einen Wert bis zu 200 GiB an, um den verfügbaren flüchtigen Speicher über den Standardwert von 20 gibibytes (GiB) für Ihre Fargate-Aufgaben zu erweitern.

12. (Optional) Um eine Datenvolumenkonfiguration für die Aufgabendefinition hinzuzufügen, wählen Sie Volume hinzufügen aus, und gehen Sie dann wie folgt vor.

    a. Geben Sie unter Volume name (Volume-Name) einen Namen für das Daten-Volume an. Der Daten-Volume-Name wird beim Erstellen eines Container-Mount-Punkts verwendet.

    b. Wählen Sie unter Volume-Konfiguration aus, ob Sie Ihr Volume bei der Erstellung der Aufgabendefinition oder während der Bereitstellung konfigurieren möchten.

    > **Note**
    >
    > Zu den Volumes, die bei der Erstellung einer Aufgabendefinition konfiguriert werden können, gehören Bind MountDocker, Amazon EFS und Amazon FSx for Windows File Server. Zu den Volumes, die bei der Bereitstellung konfiguriert werden können, wenn eine Aufgabe ausgeführt wird oder wenn ein Service erstellt oder aktualisiert wird, gehört Amazon EBS.

    c. Wählen Sie unter Volumetyp einen Volumetyp aus, der mit dem ausgewählten Konfigurationstyp kompatibel ist, und konfigurieren Sie dann den Volumetyp.

Volume-Typ	Schritte	
Halterung binden	a. Wählen Sie Add mount point (Mount-Punkt hinzufügen), und konfigurieren Sie dann Folgendes: • Wählen Sie für Container den Container für den	

Volume-Typ	Schritte	
	Bindungsbereitstellungspunkt aus. • Wählen Sie für Source volume (Quell-Volume) das Daten-Volume für die Bindungsbereitstellung im Container aus. • Geben Sie unter Container path (Container-Pfad) den Pfad zu dem Container ein, um die Bindungsbereitstellung für das Volume durchzuführen. • Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat. b. Um weitere Mount-Punkte hinzuzufügen, wählen Sie Add mount point (Mount-Punkt hinzufügen).	

Volume-Typ	Schritte	
EFS	a. Wählen Sie für File system ID (Dateisystem-ID) die Amazon-EFS-Dateisystem-ID aus. b. (Optional) Geben Sie für Root directory (Root-Verzeichnis) das Verzeichnis innerhalb des Amazon-EFS-Dateisystems ein, das als Root-Verzeichnis innerhalb des Hosts gemountet werden soll. Wenn dieser Parameter weggelassen wird, wird der Stamm des Amazon-EFS-Volumes verwendet. Lassen Sie dieses Feld leer, wenn Sie einen EFS-Zugriffspunkt verwenden möchten. c. (Optional) Wählen Sie für Access Point (Zugriffspunkt) die zu verwendende Zugriffspunkt-ID aus. d. (Optional) Um die Daten zwischen dem Amazon-EFS-Dateisystem und dem Amazon-ECS-Host zu verschlüsseln oder die Aufgabenausführung	

Volume-Typ	Schritte	
	srolle beim Mounten des Volumes zu verwenden , wählen Sie Advanced configurations (Erweiterte Konfigurationen) aus und konfigurieren Sie dann Folgendes: • Um die Daten zwischen dem Amazon-EFS-Dateisystem und dem Amazon-ECS-Host zu verschlüsseln, wählen Sie Transit encryption (Transit-Verschlüsselung) und geben Sie dann für Port den Port ein, der beim Senden verschlüsselter Daten zwischen dem Amazon-ECS-Host und dem Amazon-EFS-Server verwendet werden soll. Wenn Sie keinen Transit-Verschlüsselungsport angeben, wird die Port-Auswahlstrategie verwendet , die der Amazon EFS-Mount-Helfer verwendet. Weitere Informationen finden Sie unter EFS-Mount	

Volume-Typ	Schritte	
	-Helfer im Benutzerhandbuch für Amazon Elastic File System. • Um die in einer Aufgabendefinition definierte Amazon-ECS-Aufgaben-IAM-Rolle beim Mounten des Amazon-EFS-Dateisystems zu verwenden, wählen Sie IAM authorization (IAM-Autorisierung) aus. e. Wählen Sie Add mount point (Mount-Punkt hinzufügen), und konfigurieren Sie dann Folgendes: • Wählen Sie für Container den Container für den Bindungsbereitstellungspunkt aus. • Wählen Sie für Source volume (Quell-Volume) das Daten-Volume für die Bindungsbereitstellung im Container aus. • Geben Sie unter Container path	

Volume-Typ	Schritte	
	(Container-Pfad) den Pfad zu dem Container ein, um die Bindungsbereitstellung für das Volume durchzuführen. • Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat. f. Um weitere Mount-Punkte hinzuzufügen, wählen Sie Add mount point (Mount-Punkt hinzufügen).	

Volume-Typ	Schritte	
Docker	a. Geben Sie unter Treiber die Docker Volume-Konfiguration ein. Windows-Container unterstützen nur die Verwendung des lokalen Treibers. Um Bind-Mounts zu verwenden, geben Sie einen Host an. b. Wählen Sie für Scope (Umfang) den Volume-Lebenszyklus aus. • Damit der Lebenszyklus beim Starten und Anhalten der Aufgabe andauert, wählen Sie Task (Aufgabe). • Um das Volumen auch nach Beendigung der Aufgabe beizubehalten, wählen Sie Shared (Freigegeben). c. Wählen Sie Add mount point (Mount-Punkt hinzufügen), und konfigurieren Sie dann Folgendes: • Wählen Sie für Container den	

Volume-Typ	Schritte	
	Container für den Bindungsbereitstellungspunkt aus. • Wählen Sie für Source volume (Quell-Volume) das Daten-Volume für die Bindungsbereitstellung im Container aus. • Geben Sie unter Container path (Container-Pfad) den Pfad zu dem Container ein, um die Bindungsbereitstellung für das Volume durchzuführen. • Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat. d. Um weitere Mount-Punkte hinzuzufügen, wählen Sie Add mount point (Mount-Punkt hinzufügen).	

Volume-Typ	Schritte	
FSx for Windows File Server	a. Wählen Sie als Dateisystem-ID den Dateisystem-ID von FSx für Windows File Server aus. b. Geben Sie als Stammverzeichnis das Verzeichnis ein und geben Sie das Verzeichnis innerhalb des Dateisystems FSx for Windows File Server ein, das als Stammverzeichnis auf dem Host bereitgestellt werden soll. c. Wählen Sie für den Parameter Anmeldeinformationen aus, wie die Anmeldeinformationen gespeichert werden. • Geben Sie zur Verwendung AWS Secrets Manager den Amazon-Ressourcennamen (ARN) eines Secrets Manager Manager-Geheimnisses ein. • Geben Sie zur Verwendung AWS Systems Manager den	

Volume-Typ	Schritte	
	Amazon-Ressourcennamen (ARN) eines Systems Manager Manager-Parameters ein. d. Geben Sie für Domain den vollqualifizierten Domainnamen ein, der von einem AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) -Verzeichnis oder einem selbst gehosteten EC2-Active Directory gehostet wird. e. Wählen Sie Add mount point (Mount-Punkt hinzufügen), und konfigurieren Sie dann Folgendes: • Wählen Sie für Container den Container für den Bindungsbereitstellungspunkt aus. • Wählen Sie für Source volume (Quell-Volume) das Daten-Volume für die Bindungsbereitstellung im Container aus.	

Volume-Typ	Schritte	
	• Geben Sie unter Container path (Container-Pfad) den Pfad zu dem Container ein, um die Bindungsbereitstellung für das Volume durchzuführen. • Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat. f. Um weitere Mount-Punkte hinzuzufügen, wählen Sie Add mount point (Mount-Punkt hinzufügen).	

Volume-Typ	Schritte	
Amazon EBS	a. Wählen Sie Add mount point (Mount-Punkt hinzufügen), und konfigurieren Sie dann Folgendes: • Wählen Sie für Container den Container für den Bindungsbereitstellungspunkt aus. • Wählen Sie für Source volume (Quell-Volume) das Daten-Volume für die Bindungsbereitstellung im Container aus. • Geben Sie unter Container path (Container-Pfad) den Pfad zu dem Container ein, um die Bindungsbereitstellung für das Volume durchzuführen. • Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat. b.	

Volume-Typ	Schritte	
	Um weitere Mount-Punkte hinzuzufügen, wählen Sie Add mount point (Mount-Punkt hinzufügen).	

13. Um ein Volume aus einem anderen Container hinzuzufügen, wählen Sie Volume hinzufügen aus und konfigurieren Sie dann Folgendes:

    - Wählen Sie für Container den Container aus.
    - Wählen Sie unter Quelle den Container aus, der das Volume enthält, das Sie mounten möchten.
    - Wählen Sie für Schreibgeschützt, ob der Container nur schreibgeschützten Zugriff auf das Volume hat.

14. (Optional) Um Ihre Einstellungen für Anwendungsverfolgung und Metrikerfassung mithilfe der AWS Distro for OpenTelemetry Integration zu konfigurieren, erweitern Sie Überwachung und wählen Sie dann Metrikerfassung verwenden, um Metriken für Ihre Aufgaben zu sammeln und an Amazon CloudWatch oder Amazon Managed Service for Prometheus zu senden aus. Wenn diese Option ausgewählt ist, erstellt Amazon ECS einen AWS Distro for OpenTelemetry Container-Sidecar, der für das Senden der Anwendungsmetriken vorkonfiguriert ist. Weitere Informationen finden Sie unter Korrelieren Sie die Amazon ECS-Anwendungsleistung mithilfe von Anwendungsmetriken.

    a. Wenn Amazon ausgewählt CloudWatch ist, werden Ihre benutzerdefinierten Anwendungsmetriken CloudWatch als benutzerdefinierte Metriken weitergeleitet. Weitere Informationen finden Sie unter Anwendungsmetriken nach Amazon exportieren CloudWatch.

    > **Important**
    >
    > Wenn Sie Anwendungsmetriken nach Amazon exportieren CloudWatch, erfordert Ihre Aufgabendefinition eine Task-IAM-Rolle mit den erforderlichen Berechtigungen. Weitere Informationen finden Sie unter Erforderliche IAM-

Berechtigungen für AWS Distro für die OpenTelemetry Integration mit Amazon CloudWatch.

b. Wenn Sie Amazon Managed Service for Prometheus (Prometheus libraries instrumentation) (Amazon Managed Service for Prometheus (Instrumentierung der Prometheus-Bibliotheken)) ausgewählt ist, werden Ihre CPU-, Arbeitsspeicher-, Netzwerk- und Speichermetriken auf Aufgabenebene sowie Ihre benutzerdefinierten Anwendungsmetriken an Amazon Managed Service for Prometheus weitergeleitet. Geben Sie für Workspace Remote Write Endpoint die URL des Remote-Write-Endpunkts für Ihren Prometheus Workspace ein. Geben Sie für das Scraping-Ziel den Host und den Port ein, über den der AWS Distro for OpenTelemetry Collector nach Metrikdaten suchen kann. Weitere Informationen finden Sie unter Exportieren von Anwendungsmetriken an Amazon Managed Service for Prometheus.

> **Important**
>
> Beim Exportieren von Anwendungsmetriken nach Amazon Managed Service for Prometheus erfordert Ihre Aufgabendefinition eine Aufgaben-IAM-Rolle Aufgabe mit den erforderlichen Berechtigungen. Weitere Informationen finden Sie unter Erforderliche IAM-Berechtigungen für AWS Distro für die OpenTelemetry Integration mit Amazon Managed Service for Prometheus.

c. Wenn Sie Amazon Managed Service for Prometheus (OpenTelemetry Instrumentierung) auswählen, werden Ihre CPU-, Arbeitsspeicher-, Netzwerk- und Speichermetriken auf Taskebene sowie Ihre benutzerdefinierten Anwendungsmetriken an Amazon Managed Service for Prometheus weitergeleitet. Geben Sie für Workspace Remote Write Endpoint die URL des Remote Write-Endpunkts für Ihren Workspace ein. Prometheus Weitere Informationen finden Sie unter Exportieren von Anwendungsmetriken an Amazon Managed Service for Prometheus.

> **Important**
>
> Beim Exportieren von Anwendungsmetriken nach Amazon Managed Service for Prometheus erfordert Ihre Aufgabendefinition eine Aufgaben-IAM-Rolle Aufgabe mit den erforderlichen Berechtigungen. Weitere Informationen finden Sie unter Erforderliche IAM-Berechtigungen für AWS Distro für die OpenTelemetry Integration mit Amazon Managed Service for Prometheus.

15. (Optional) Erweitern Sie den Abschnitt Tags zum Hinzufügen von Tags als Schlüssel-Wert-Paare zur Aufgabendefinition.

    - [Ein Tag hinzufügen] Wählen Sie Add tag (Tag hinzufügen) und führen Sie dann das Folgende aus:
      - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
      - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
    - [Tag entfernen] Wählen Sie neben dem Tag die Option Remove tag (Tag löschen) aus.
16. Wählen Sie Erstellen, um die Aufgabendefinition zu registrieren.

Amazon ECS console JSON editor

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie im Menü Neue Aufgabendefinition erstellen die Option Neue Aufgabendefinition mit JSON erstellen aus.
4. Bearbeiten Sie im JSON-Editorfeld Ihre JSON-Datei,

   Der JSON muss die in the section called “JSON-Validierung” angegebenen Validierungsprüfungen bestehen.
5. Wählen Sie Erstellen.

Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole

Eine Revision der Aufgabendefinition ist eine Kopie der aktuellen Aufgabendefinition mit den neuen Parameterwerten, die die vorhandenen ersetzen. Alle Parameter, die Sie nicht ändern, befinden sich in der neuen Revision.

Um eine Aufgabendefinition zu aktualisieren, erstellen Sie eine Revision der Aufgabendefinition. Wenn die Aufgabendefinition in einem Service verwendet wird, aktualisieren diesen Service, sodass er die aktualisierte Aufgabendefinition verwendet.

Wenn Sie eine Revision erstellen, können Sie die folgenden Containereigenschaften und Umgebungseigenschaften ändern.

- Container-Image-URI
- Port-Zuweisungen
- Umgebungsvariablen
- Aufgabengröße
- Größe des Containers
- Aufgabenrolle
- Aufgabenausführungsrolle
- Volumen und Befestigungspunkte für Container
- Private Registrierung

JSON-Validierung

Der JSON-Editor der Amazon-ECS-Konsole validiert Folgendes in der JSON-Datei:

- Die Datei ist eine gültige JSON-Datei
- Die Datei enthält keine irrelevanten Schlüssel
- Die Datei enthält den `familyName`-Parameter
- Unter `containerDefinitions` ist mindestens ein Eintrag vorhanden

Verfahren

Amazon ECS console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre Aufgabendefinition enthalten ist.
3. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
4. Wählen Sie die Aufgabendefinition.
5. Markieren Sie die Aufgabendefinitionsversion und wählen Sie dann Neue Version erstellen, Neue Version erstellen.
6. Nehmen Sie auf der Seite Create new task definition revision (Neue Revision der Aufgabendefinition erstellen) die Änderungen vor. Wenn Sie beispielsweise die vorhandenen Containerdefinitionen ändern möchten (wie z. B. das Container-Image, die Arbeitsspeicher-

Limits oder Port-Mappings), wählen Sie den Container aus, um die Änderungen vorzunehmen.

7. Verifizieren Sie die Informationen und wählen Sie dann Aktualisieren.
8. Wenn Ihre Aufgabendefinition in einem Service verwendet wird, aktualisieren Ihren Service durch die aktualisierte Aufgabendefinition. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.

Amazon ECS console JSON editor

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new revision (Neue Revision erstellen), Create new revision with JSON (Neue Revision mit JSON) erstellen.
4. Bearbeiten Sie im JSON-Editorfeld Ihre JSON-Datei,

   Der JSON muss die in the section called “JSON-Validierung” angegebenen Validierungsprüfungen bestehen.
5. Wählen Sie Erstellen.

Abmeldung einer Revision der Amazon ECS-Aufgabendefinition mithilfe der Konsole

Wenn Sie eine bestimmte Aufgabendefinitionsrevision in Amazon ECS nicht mehr benötigen, können Sie die Revision der Aufgabendefinition deregistrieren, sodass sie nicht mehr in Ihren `ListTaskDefinition` API-Aufrufen oder in der Konsole angezeigt wird, wenn Sie eine Aufgabe ausführen oder einen Service aktualisieren möchten.

Wenn Sie die Registrierung einer Revision einer Aufgabendefinition aufheben, wird diese sofort als `INACTIVE` gekennzeichnet. Vorhandene Aufgaben und Dienste, die auf eine `INACTIVE`-Taskdefinitionsrevision verweisen werden weiterhin ohne Unterbrechung ausgeführt. Bestehende Dienste, die auf eine `INACTIVE`-Revision der Aufgabendefinition verweisen, können durch Änderung der gewünschten Anzahl des Service weiterhin nach oben oder unten skaliert werden.

Sie können keine `INACTIVE`-Taskdefinitionsrevision verwenden, um neue Aufgaben auszuführen oder neue Dienste zu erstellen. Sie können einen vorhandenen Service auch nicht dahingehend

aktualisieren, dass er auf eine `INACTIVE`-Revision einer Aufgabendefinition verweist (Obwohl nach Aufheben der Registrierung ein Zeitfenster von bis zu 10 Minuten bestehen kann, in dem diese Einschränkungen noch nicht wirksam sind).

> **Note**
>
> Wenn Sie die Registrierung aller Revisionen in einer Aufgabenfamilie aufheben, wird die Aufgabendefinitionsfamilie in die `INACTIVE`-Liste verschoben. Das Hinzufügen einer neuen Version einer `INACTIVE`-Aufgabendefinition verschiebt die Aufgabendefinitionsfamilie zurück in die `ACTIVE`-Liste.
> Zu diesem Zeitpunkt bleiben Überarbeitungen von `INACTIVE`-Aufgabendefinitionen in Ihrem Konto unbegrenzt erkennbar. Dieses Verhalten kann sich jedoch in Zukunft ändern. Daher sollten Sie sich nicht darauf verlassen, dass Überarbeitungen von `INACTIVE`-Aufgabendefinitionen über den Lebenszyklus der zugeordneten Aufgaben und Services hinaus bestehen.

AWS CloudFormation Stapel

Das folgende Verhalten gilt für Aufgabendefinitionen, die vor dem 12. Januar 2023 in der neuen Amazon ECS-Konsole erstellt wurden.

Wenn Sie eine Aufgabendefinition erstellen, erstellt die Amazon ECS-Konsole automatisch einen CloudFormation Stack, dessen Name mit beginnt`ECS-Console-V2-TaskDefinition-`. Wenn Sie das AWS CLI oder ein AWS SDK verwendet haben, um die Registrierung der Aufgabendefinition aufzuheben, müssen Sie den Aufgabendefinitionsstapel manuell löschen. Weitere Informationen finden Sie im AWS CloudFormation Benutzerhandbuch unter Löschen eines Stacks.

Für Aufgabendefinitionen, die nach dem 12. Januar 2023 erstellt wurden, wird kein CloudFormation Stapel automatisch für sie erstellt.

Verfahren

So melden Sie eine neue Aufgabendefinition ab (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre Aufgabendefinition enthalten ist.
3. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.

4. Wählen Sie auf der Seite Task Definitions (Aufgabendefinitionen) die Aufgabendefinitionsfamilie aus, die eine oder mehrere Revisionen enthält, deren Registrierung Sie aufheben möchten.
5. Wählen Sie auf der Seite mit dem Namen der Aufgabendefinition die zu löschenden Revisionen aus, und klicken Sie dann auf Aktionen, Registrierung aufheben.
6. Überprüfen Sie die Informationen im Fenster Deregister (Abmelden) und wählen Sie dann Deregister (Abmelden), um den Vorgang abzuschließen.

Löschen einer Revision der Amazon ECS-Aufgabendefinition mithilfe der Konsole

Wenn Sie eine bestimmte Aufgabendefinitionsrevision in Amazon ECS nicht mehr benötigen, können Sie die Version der Aufgabendefinition löschen.

Wenn Sie eine Aufgabendefinitionsrevision löschen, wechselt sie sofort von `INACTIVE` zu `DELETE_IN_PROGRESS`. Vorhandene Aufgaben und Dienste, die auf eine `DELETE_IN_PROGRESS`-Aufgabendefinitionsrevision verweisen werden weiterhin ohne Unterbrechung ausgeführt.

Sie können keine `DELETE_IN_PROGRESS`-Aufgabendefinitionsrevision verwenden, um neue Aufgaben auszuführen oder neue Dienste zu erstellen. Sie können einen vorhandenen Service auch nicht dahingehend aktualisieren, dass er auf eine `DELETE_IN_PROGRESS`-Revision einer Aufgabendefinition verweist.

Wenn Sie alle Versionen der `INACTIVE`-Aufgabendefinition löschen, wird der Name der Aufgabendefinition nicht in der Konsole angezeigt und nicht in der API zurückgegeben. Wenn sich eine Version der Aufgabendefinition im `DELETE_IN_PROGRESS` Status befindet, wird der Name der Aufgabendefinition in der Konsole angezeigt und in der API zurückgegeben. Der Name der Aufgabendefinition wird von Amazon ECS beibehalten und die Version wird erhöht, wenn Sie das nächste Mal eine Aufgabendefinition mit diesem Namen erstellen.

Amazon-ECS-Ressourcen, die einen Löschvorgang blockieren können

Eine Anforderung zum Löschen von Aufgabendefinitionen wird nicht abgeschlossen, wenn Amazon ECS-Ressourcen vorhanden sind, die von der Revision der Aufgabendefinition abhängen. Die folgenden Ressourcen können verhindern, dass eine Aufgabendefinition gelöscht wird:

- Amazon-ECS-Aufgaben – Die Aufgabendefinition ist erforderlich, damit die Aufgabe fehlerfrei bleibt.

- Amazon-ECS-Bereitstellungen und Aufgabensätze – Die Aufgabendefinition ist erforderlich, wenn ein Skalierungsereignis für eine Amazon-ECS-Bereitstellung oder einen Aufgabensatz ausgelöst wird.

Bleibt Ihre Aufgabendefinition unverändert, `DELETE_IN_PROGRESS` können Sie die Konsole oder die verwenden, AWS CLI um die Ressourcen zu identifizieren und dann zu beenden, die das Löschen der Aufgabendefinition blockieren.

Löschen der Aufgabendefinition, nachdem die blockierende Ressource entfernt wurde

Die folgenden Regeln gelten, nachdem Sie die Ressourcen entfernt haben, die das Löschen der Aufgabendefinition blockieren:

- Amazon-ECS-Aufgaben – Nach dem Beenden der Aufgabe kann es bis zu 1 Stunde dauern, bis das Löschen der Aufgabendefinition abgeschlossen ist.
- Amazon-ECS-Bereitstellungen und Aufgabensätze – Es kann bis zu 24 Stunden dauern, bis das Löschen der Aufgabendefinition abgeschlossen ist, nachdem die Bereitstellung oder der Aufgabensatz gelöscht wurde.

Verfahren

So löschen Sie Aufgabendefinitionen (Amazon-ECS-Konsole)

Sie müssen die Registrierung einer Revision der Aufgabendefinition aufheben, bevor Sie sie löschen. Weitere Informationen finden Sie unter the section called “Abmelden einer Revision einer Aufgabendefinition mithilfe der Konsole”.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre Aufgabendefinition enthalten ist.
3. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
4. Wählen Sie auf der Seite Aufgabendefinitionen die Aufgabendefinitionsfamilie aus, die eine oder mehrere Revisionen enthält, die Sie löschen möchten.
5. Wählen Sie auf der Seite mit dem Namen der Aufgabendefinition die zu löschenden Versionen aus, und klicken Sie dann auf Aktionen, Löschen.

   Wenn Löschen nicht verfügbar ist, müssen Sie die Registrierung der Aufgabendefinition aufheben.

6. Überprüfen Sie die Informationen im Bestätigungsfeld Löschen, und wählen Sie dann Löschen, um den Vorgang abzuschließen.

Anwendungsfälle für Amazon ECS-Aufgabendefinitionen

Erfahren Sie mehr darüber, wie Sie Aufgabendefinitionen für verschiedene AWS Dienste und Funktionen schreiben.

Abhängig von Ihrer Arbeitslast müssen bestimmte Aufgabendefinitionsparameter festgelegt werden. Außerdem müssen Sie für den EC2-Starttyp bestimmte Instances auswählen, die für den Workload entwickelt wurden.

Themen

Amazon ECS-Aufgabendefinitionen für GPU-Workloads

Amazon ECS unterstützt Workloads, die GPUs verwenden, wenn Sie Cluster mit Container-Instances erstellen, die GPUs unterstützen. GPU-basierte Amazon-EC2-Container-Instances, die die Instance-Typen p2, p3, p5, g3, g4 und g5 verwenden, bieten Zugriff auf NVIDIA-GPUs. Weitere Informationen finden Sie unter Linux Accelerated Computing Instances im Amazon EC2 EC2-Benutzerhandbuch.

Amazon ECS bietet eine GPU-optimierte AMI, die mit vorkonfigurierten NVIDIA-Kernel-Treibern und einer Docker-GPU-Laufzeitumgebung ausgestattet ist. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Sie können in Ihrer Aufgabendefinition eine Reihe von GPUs für die Platzierung von Aufgaben auf Container-Ebene festlegen. Amazon ECS plant verfügbare GPU-fähige Container-Instances und heftet physische GPUs für eine optimale Leistung an geeignete Container an.

Die folgenden GPU-basierten Instance-Typen von Amazon EC2 werden unterstützt. Weitere Informationen finden Sie unter Amazon EC2 P2-Instances, Amazon EC2 P3-Instances, Amazon EC2 P4d-Instances, Amazon EC2 P5-Instances, Amazon EC2 G3-Instances, Amazon EC2 G4-Instances, Amazon EC2 G5-Instancesund Amazon EC2 G6-Instances.

Instance-Typ	GPUs	GPU-Speicher (GiB)	vCPUs	Arbeitsspeicher (GiB)
p3.2xgroß	1	16	8	61
p3.8xgroß	4	64	32	244
p3.16xgroß	8	128	64	488
p3dn.24xgroß	8	256	96	768
p4d.24xgroß	8	320	96	1 152
p5.48xlarge	8	640	192	2048
g3s.xgroß	1	8	4	30,5
g3.4xgroß	1	8	16	122
g3.8xgroß	2	16	32	244
g3.16xgroß	4	32	64	488
g4dn.xgroß	1	16	4	16
g4dn.2xgroß	1	16	8	32
g4dn.4xgroß	1	16	16	64
g4dn.8xgroß	1	16	32	128
g4dn.12xgroß	4	64	48	192

Instance-Typ	GPUs	GPU-Speicher (GiB)	vCPUs	Arbeitsspeicher (GiB)
g4dn.16xgroß	1	16	64	256
g5.xgroß	1	24	4	16
g5.2xlarge	1	24	8	32
g5.4xlarge	1	24	16	64
g5.8xlarge	1	24	32	128
g5.16xlarge	1	24	64	256
g5.12xlarge	4	96	48	192
g5.24xlarge	4	96	96	384
g5.48xlarge	8	192	192	768
g6.xlarge	1	24	4	16
g 6.2 x groß	1	24	8	32
g 6,4 x groß	1	24	16	64
g 6,8 x groß	1	24	32	128
g 6.16.x groß	1	24	64	256
g 6.12 x groß	4	96	48	192
g 6.24 x groß	4	96	48	192
g 6,48 x groß	8	192	192	768
g6. Metall	8	192	192	768
gr 6.4 x groß	1	24	16	128
gr 6,8 x groß	1	24	32	256

Sie können die Amazon Machine Image (AMI) -ID für Amazon ECS-optimierte AMIs abrufen, indem Sie die AWS Systems Manager Parameter Store-API abfragen. Mit diesem Parameter müssen Sie Amazon-ECS-optimierte AMI-IDs nicht manuell nachschlagen. Weitere Informationen zur Systems Manager Parameter Store-API finden Sie unter GetParameter. Der von Ihnen verwendete Benutzer muss über die `ssm:GetParameter`-IAM-Berechtigung zum Abrufen der Amazon-EKS-optimierten AMI-Metadaten verfügen.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

Überlegungen

> **Note**
>
> Die Unterstützung für den g2-Instance-Familientyp ist veraltet.
> Der Instance-Familientyp p2 wird nur von Versionen vor `20230912` des Amazon-ECS-GPU-optimierten AMI unterstützt. Wenn Sie weiterhin P2-Instances verwenden müssen, finden Sie weitere Informationen unter Was tun, wenn Sie eine P2-Instance benötigen.
> Direkte Updates der NVIDIA/CUDA-Treiber auf diesen beiden Instance-Familientypen können zu potenziellen Ausfällen bei GPU-Workloads führen.

Wir empfehlen, Folgendes zu beachten, bevor Sie auf Amazon ECS mit GPUs arbeiten.

- Ihre Cluster können eine Mischung aus GPU- und Nicht-GPU-Container-Instances enthalten.
- Sie können GPU-Workloads auf externen Instances ausführen. Stellen Sie beim Anmelden einer externen Instance bei Ihrem Cluster sicher, dass das `--enable-gpu`-Flag im Installationsskript enthalten ist. Weitere Informationen finden Sie unter Registrierung einer externen Instance in einem Amazon ECS-Cluster.
- Sie müssen in Ihrer Agenten-Konfigurationsdatei `ECS_ENABLE_GPU_SUPPORT` auf `true` setzen. Weitere Informationen finden Sie unter the section called “Container-Agent-Konfiguration”.
- Wenn Sie eine Aufgabe ausführen oder einen Service erstellen, können Sie beim Konfigurieren der Platzierungsbedingungen mithilfe von Attributen des Instance-Typs festlegen, für welche Ihrer Container-Instances die Aufgabe gestartet wird. Auf diese Weise können Sie Ihre Ressourcen effektiver verwenden. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

Das folgende Beispiel startet eine Aufgabe auf einer `g4dn.xlarge`-Container-Instance in Ihrem Standard-Cluster.

```
aws ecs run-task --cluster default --task-definition ecs-gpu-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
  g4dn.xlarge" --region us-east-2
```

- Für jeden Container, der eine in der Container-Definition festgelegte GPU-Ressourcenanforderung hat, legt Amazon ECS fest, dass die Container-Laufzeitumgebung die NVIDIA-Container-Laufzeitumgebung ist.
- Die NVIDIA Container-Laufzeitumgebung erfordert, dass einige Umgebungsvariablen im Container festgelegt werden, damit sie korrekt funktioniert. Eine Liste dieser Umgebungsvariablen finden Sie unter Spezialisierte Konfigurationen mit Docker. Amazon ECS legt die `NVIDIA_VISIBLE_DEVICES`-Umgebungsvariablen als Liste der GPU-Geräte-IDs, die Amazon ECS dem Container zuweist, fest. Die anderen erforderlichen Umgebungsvariablen legt Amazon ECS nicht fest. Stellen Sie daher sicher, dass Ihr Container-Image sie festlegt oder dass sie in der Container-Definition festgelegt sind.
- Die p5-Instance-Typ-Familie wird auf Version `20230929` und höher des Amazon-ECS GPU-optimierten AMI unterstützt.
- Die g4-Instance-Typ-Familie wird auf Version `20230913` und höher des Amazon ECS GPU-optimierten AMI unterstützt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs. Sie wird im Workflow „Cluster erstellen“ in der Amazon-ECS-Konsole nicht unterstützt. Um diese Instance-Typen zu verwenden, müssen Sie entweder die Amazon EC2 EC2-Konsole oder die API verwenden und die Instances manuell in Ihrem Cluster registrieren. AWS CLI
- Der Instance-Typ p4d.24xlarge funktioniert nur mit CUDA 11 oder höher.
- Das für Amazon ECS GPU-optimierte AMI ist IPv6 aktiviert, was zu Problemen bei der Verwendung von `yum` führt. Dies kann durch Konfigurieren von `yum` zur Verwendung von IPv4 mit dem folgenden Befehl gelöst werden.

```
echo "ip_resolve=4" >> /etc/yum.conf
```

- Wenn Sie ein Container-Image erstellen, das die NVIDIA/CUDA-Basis-Images nicht verwendet, müssen Sie die Container-Laufzeit-Variable `NVIDIA_DRIVER_CAPABILITIES` auf einen der folgenden Werte festlegen:
  - `utility,compute`
  - `all`

Informationen zum Festlegen der Variablen finden Sie unter Steuern der NVIDIA Container Runtime auf der NVIDIA-Website.

- GPUs werden in Windows-Containern nicht unterstützt.

Starten Sie eine GPU-Container-Instance für Amazon ECS

Um eine GPU-Instance auf Amazon ECS zu verwenden, müssen Sie eine Startvorlage und eine Benutzerdatendatei erstellen und die Instance starten.

Anschließend können Sie eine Aufgabe ausführen, die eine für GPU konfigurierte Aufgabendefinition verwendet.

Verwenden einer Startvorlage

Sie können eine Startvorlage erstellen.

- Erstellen Sie eine Startvorlage, die die Amazon ECS-optimierte GPU-AMI-ID für das AMI verwendet. Informationen zum Erstellen einer Startvorlage finden Sie unter Erstellen einer neuen Startvorlage mithilfe von Parametern, die Sie im Amazon EC2 EC2-Benutzerhandbuch definieren.

  Verwenden Sie die AMI-ID aus dem vorherigen Schritt für das Amazon Machine-Image. Informationen zur Angabe der AMI-ID mit dem Systems Manager Manager-Parameter finden Sie unter Spezifizieren eines Systems Manager Manager-Parameters in einer Startvorlage im Amazon EC2 EC2-Benutzerhandbuch.

  Fügen Sie den Benutzerdaten in der Startvorlage Folgendes hinzu. Ersetzen Sie *cluster-name* durch den Namen Ihres Clusters.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

Verwenden Sie AWS CLI

Sie können den verwenden AWS CLI , um die Container-Instance zu starten.

1. Erstellen Sie eine Datei mit dem Namen `userdata.toml`. Diese Datei wird für Instance-Benutzerdaten verwendet. Ersetzen Sie *`cluster-name`* durch den Namen Ihres Clusters.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

2. Führen Sie den folgenden Befehl aus, um die GPU-AMI-ID abzurufen. Sie verwenden dies im folgenden Schritt.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

3. Führen Sie den folgenden Befehl aus, um die GPU-Instanz zu starten. Denken Sie daran, die folgenden Parameter zu ersetzen:

   - Ersetzen Sie das *`Subnetz`* durch die ID des privaten oder öffentlichen Subnetzes, in dem Ihre Instance gestartet wird.
   - Ersetzen Sie *`gpu_ami`* durch die AMI-ID aus dem vorherigen Schritt.
   - Ersetzen Sie *`t3.large`* durch den Instance-Typ, den Sie verwenden möchten.
   - Ersetzen Sie *`Region`* durch den Regionscode.

```
aws ec2 run-instances --key-name ecs-gpu-example \
   --subnet-id subnet \
   --image-id gpu_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications 'ResourceType=instance,Tags=[{Key=GPU,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob die Container-Instance im Cluster registriert ist. Denken Sie beim Ausführen dieses Befehls daran, die folgenden Parameter zu ersetzen:

   - Ersetzen Sie *`cluster`* durch Ihren Cluster-Namen.
   - Ersetzen Sie *`region`* durch Ihren Regionalcode.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Angeben von GPUs in einer Amazon ECS-Aufgabendefinition

Um die GPUs für eine Container-Instance und die Docker-GPU-Laufzeitumgebung zu nutzen, müssen Sie in der Aufgabendefinition die für Ihren Container erforderliche Anzahl von GPUs angeben. Wenn GPU-fähige Container platziert werden, heftet der Amazon-ECS-Container-Agent die gewünschte Anzahl von physischen GPUs an den entsprechenden Container an. Die Anzahl der für alle Container in einer Aufgabe reservierten GPUs darf nicht größer sein als die Anzahl der verfügbaren GPUs für die Container-Instance, für die die Aufgabe gestartet wird. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

> **Important**
>
> Wenn Ihre GPU-Anforderungen in der Aufgabendefinition nicht angegeben werden, verwendet die Aufgabe die Standard-Docker-Laufzeit.

Das folgende Beispiel zeigt das JSON-Format für die GPU-Anforderungen in einer Aufgabendefinition:

```
{
  "containerDefinitions": [
     {
        ...
        "resourceRequirements" : [
            {
               "type" : "GPU",
               "value" : "2"
            }
        ],
     },
...
}
```

Das folgende Beispiel veranschaulicht die Syntax eines Docker-Containers, der eine GPU-Anforderung angibt. Dieser Container verwendet 2 GPUs, führt das Dienstprogramm `nvidia-smi` aus und wird dann beendet.

```
{
  "containerDefinitions": [
    {
      "memory": 80,
      "essential": true,
      "name": "gpu",
      "image": "nvidia/cuda:11.0.3-base",
      "resourceRequirements": [
         {
           "type":"GPU",
           "value": "2"
         }
      ],
      "command": [
        "sh",
        "-c",
        "nvidia-smi"
      ],
      "cpu": 100
    }
  ],
  "family": "example-ecs-gpu"
}
```

Was tun, wenn Sie eine P2-Instance benötigen

Wenn Sie die P2-Instance verwenden müssen, können Sie eine der folgenden Optionen verwenden, um die Instances weiterhin zu verwenden.

Sie müssen die Benutzerdaten der Instance für beide Optionen ändern. Weitere Informationen finden Sie unter Arbeiten mit Instance-Benutzerdaten im Amazon EC2 EC2-Benutzerhandbuch.

Verwenden der zuletzt unterstützten GPU-optimierten AMI

Sie können die `20230906`-Version des GPU-optimierten AMI verwenden und den Instance-Benutzerdaten Folgendes hinzufügen.

Ersetzen Sie cluster-name durch den Namen Ihres Clusters.

```
#!/bin/bash
echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Die aktuellsten GPU-optimierten AMI verwenden und die Benutzerdaten aktualisieren

Sie können den Benutzerdaten der Instance Folgendes hinzufügen. Dadurch werden die Nvidia 535/Cuda12.2-Treiber deinstalliert und anschließend die Nvidia 470/Cuda11.4-Treiber installiert und die Version repariert.

```
#!/bin/bash
yum remove -y cuda-toolkit* nvidia-driver-latest-dkms*
tmpfile=$(mktemp)
cat >$tmpfile <<EOF
[amzn2-nvidia]
name=Amazon Linux 2 Nvidia repository
mirrorlist=\$awsproto://\$amazonlinux.\$awsregion.\$awsdomain/\$releasever/amzn2-
nvidia/latest/\$basearch/mirror.list
priority=20
gpgcheck=1
gpgkey=https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/7fa2af80.pub
enabled=1
exclude=libglvnd-*
EOF

mv $tmpfile /etc/yum.repos.d/amzn2-nvidia-tmp.repo
yum install -y system-release-nvidia cuda-toolkit-11-4 nvidia-driver-latest-
dkms-470.182.03
yum install -y libnvidia-container-1.4.0 libnvidia-container-tools-1.4.0 nvidia-
container-runtime-hook-1.4.0 docker-runtime-nvidia-1

echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
nvidia-smi
```

Ihr eigenes P2-kompatibles und GPU-optimiertes AMI erstellen

Sie können Ihr eigenes benutzerdefiniertes GPU-optimiertes Amazon-ECS-AMI erstellen, das mit P2-Instances kompatibel ist, und dann P2-Instances mithilfe des AMI starten.

1. Führen Sie den folgenden Befehl aus, um das `amazon-ecs-ami repo` zu klonen.

```
git clone https://github.com/aws/amazon-ecs-ami
```

2. Stellen Sie die erforderlichen Amazon-ECS-Agent- und Amazon-Linux-AMI-Quellversionen in `release.auto.pkrvars.hcl` oder `overrides.auto.pkrvars.hcl` ein.
3. Führen Sie den folgenden Befehl aus, um ein privates P2-kompatibles EC2-AMI zu erstellen.

   Ersetzen Sie Region durch die Region mit der Instance-Region.

   ```
   REGION=region make al2keplergpu
   ```

4. Verwenden Sie das AMI mit den folgenden Instance-Benutzerdaten, um eine Verbindung zum Amazon-ECS-Cluster herzustellen.

   Ersetzen Sie cluster-name durch den Namen Ihres Clusters.

   ```
   #!/bin/bash
   echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
   ```

Amazon ECS-Aufgabendefinitionen für Videotrankodierungs-Workloads

Um Workloads zur Video-Transcodierung auf Amazon ECS zu verwenden, registrieren Sie die Amazon-EC2-VT1-Instances. Nachdem Sie diese Instances registriert haben, können Sie Live- und vorab gerenderte Video-Transcodierungs-Workloads als Aufgaben auf Amazon ECS ausführen. Amazon-EC2-VT1-Instances verwenden Xilinx U30-Medientranscodierungskarten, um Live- und vorab gerenderte Video-Transcodierungs-Workloads zu beschleunigen.

> **Note**
>
> Anleitungen für das Ausführen von Workloads zur Video-Transcodierung in anderen Containern als Amazon ECS finden Sie in der Xilinx-Dokumentation.

Überlegungen

Seien Sie sich der folgenden Überlegungen bewusst, bevor Sie mit der Bereitstellung von VT1 auf Amazon ECS beginnen:

- Ihre Cluster können eine Mischung aus VT1- und Nicht-VT1-Instances enthalten.

- Sie benötigen eine Linux-Anwendung, die Xilinx U30-Medientranscodierungskarten mit beschleunigten H.264/AVC- und H.265/HEVC-Codecs verwendet.

  > **Important**
  >
  > Anwendungen, die andere Codecs verwenden, haben möglicherweise keine verbesserte Leistung auf VT1-Instances.

- Nur eine Transcodierungsaufgabe kann auf einer U30-Karte ausgeführt werden. Jede Karte hat zwei Geräte, die damit verbunden sind. Sie können so viele Transcodierungsaufgaben ausführen, wie es Karten für jede Ihrer VT1-Instance gibt.
- Wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen, können Sie beim Konfigurieren der Aufgabenplatzierungbedingungen Attribute des Instance-Typs verwenden. Dadurch wird sichergestellt, dass die Aufgabe auf der von Ihnen angegebenen Container-Instance gestartet wird. So können Sie Ihre Ressourcen effektiv nutzen und sicherstellen, dass sich Ihre Aufgaben für Video-Transcodierungs-Workloads auf Ihren VT1-Instances befinden. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

  Im folgenden Beispiel wird eine Aufgabe auf einer `vt1.3xlarge`-Instance auf Ihrem `default`-Cluster ausgeführt.

  ```
  aws ecs run-task \
       --cluster default \
       --task-definition vt1-3xlarge-xffmpeg-processor \
       --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
   vt1.3xlarge"
  ```

- Sie können einen Container so konfigurieren, dass er die spezifische U30-Karte verwendet, die auf der Host-Container-Instance verfügbar ist. Sie können dazu den `linuxParameters`-Parameter nutzen und die Geräteinformationen angeben. Weitere Informationen finden Sie unter Anforderungen an die Aufgabendefinition.

Verwenden eines VT1-AMI

Sie haben zwei Möglichkeiten, ein AMI auf Amazon EC2 für Amazon-ECS-Container-Instances auszuführen. Die erste Option besteht darin, das offizielle Xilinx-AMI auf AWS Marketplace zu verwenden. Die zweite Option besteht darin, Ihr eigenes AMI aus dem Beispiel-Repository zu erstellen.

- Xilinx bietet AMIs auf dem. AWS Marketplace
- Amazon ECS stellt ein Beispiel-Repository bereit, mit dem Sie ein AMI für Video-Transcodierungs-Workloads erstellen können. Dieses AMI wird mit Xilinx U30-Treibern geliefert. Sie finden das Repository, das Packer-Skripte enthält, auf. GitHub Weitere Informationen zu Packer finden Sie in der Packer-Dokumentation.

Anforderungen an die Aufgabendefinition

Um Video-Transcodierungscontainer auf Amazon ECS auszuführen, muss Ihre Aufgabendefinition eine Video-Transcodierungsanwendung enthalten, die die beschleunigten H.264/AVC- und H.265/HEVC-Codecs verwendet. Sie können ein Container-Image erstellen, indem Sie den Schritten auf dem GitHubXilinx folgen.

Die Aufgabendefinition muss für den Instance-Typ spezifisch sein. Die Instance-Typen sind 3xlarge, 6xlarge und 24xlarge. Sie müssen einen Container so konfigurieren, dass er spezifische Xilinx U30-Geräte verwendet, die auf der Host-Container-Instance verfügbar sind. Dazu können Sie den `linuxParameters`-Parameter verwenden. In der folgenden Tabelle sind die Karten und Geräte aufgeführt SoCs , die für jeden Instance-Typ spezifisch sind.

Instance-Typ	vCPUs	RAM (GiB)	U30 Accelerator-Karten	Adressierbare XCU30 SoC-Geräte	Gerätepfade
vt1.3xlarge	12	24	1	2	`/dev/dri/renderD128 ,/dev/dri/renderD129`
vt1.6xlarge	24	48	2	4	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/`

Instance-Typ	vCPUs	RAM (GiB)	U30 Accelerator-Karten	Adressierbare XCU30 SoC-Geräte	Gerätepfade
					`renderD130 ,/dev/dri/renderD131`

Instance-Typ	vCPUs	RAM (GiB)	U30 Accelerator-Karten	Adressierbare XCU30 SoC-Geräte	Gerätepfade
vt1.24xlarge	96	182	8	16	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131 ,/dev/dri/renderD132 ,/dev/dri/renderD133 ,/dev/dri/renderD134 ,/dev/dri/renderD135 ,/dev/dri/renderD136 ,/dev/dri/renderD137 ,/dev/dri/renderD13`

Instance-Typ	vCPUs	RAM (GiB)	U30 Accelerator-Karten	Adressierbare XCU30 SoC-Geräte	Gerätepfade
					8 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/dri/renderD143

Important

Wenn die Aufgabendefinition Geräte auflistet, die die EC2-Instance nicht hat, kann die Aufgabe nicht ausgeführt werden. Wenn die Aufgabe fehlschlägt, wird die folgende Fehlermeldung in `stoppedReason` angezeigt: `CannotStartContainerError: Error response from daemon: error gathering device information while adding custom device "/dev/dri/renderD130": no such file or directory.`

Angeben der Videotranskodierung in einer Amazon ECS-Aufgabendefinition

Im folgenden Beispiel sehen Sie die Syntax, die für eine Aufgabendefinition eines Linux-Containers auf Amazon EC2 verwendet wird. Diese Aufgabendefinition gilt für Container-Images, die gemäß dem Verfahren erstellt werden, das in der Xilinx-Dokumentation beschrieben ist. Wenn Sie dieses Beispiel

verwenden, ersetzen Sie `image` durch Ihr eigenes Image und kopieren Sie Ihre Videodateien in die Instance im `/home/ec2-user`-Verzeichnis.

vt1.3xlarge

1. Erstellen Sie eine Textdatei mit dem Namen `vt1-3xlarge-ffmpeg-linux.json` und dem folgenden Inhalt.

```
{
    "family": "vt1-3xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.3xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
```

```
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registrieren Sie die Aufgabendefinition.

```
aws ecs register-task-definition --family vt1-3xlarge-xffmpeg-processor --cli-
input-json file://vt1-3xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.6xlarge

1. Erstellen Sie eine Textdatei mit dem Namen `vt1-6xlarge-ffmpeg-linux.json` und dem folgenden Inhalt.

```
{
    "family": "vt1-6xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
```

```
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.6xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
```

```
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registrieren Sie die Aufgabendefinition.

```
aws ecs register-task-definition --family vt1-6xlarge-xffmpeg-processor --cli-
input-json file://vt1-6xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.24xlarge

1. Erstellen Sie eine Textdatei mit dem Namen `vt1-24xlarge-ffmpeg-linux.json` und dem folgenden Inhalt.

```
{
    "family": "vt1-24xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
```

```
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.24xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
```

```
                    "containerPath": "/dev/dri/renderD131",
                    "hostPath": "/dev/dri/renderD131",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD132",
                    "hostPath": "/dev/dri/renderD132",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD133",
                    "hostPath": "/dev/dri/renderD133",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD134",
                    "hostPath": "/dev/dri/renderD134",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD135",
                    "hostPath": "/dev/dri/renderD135",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD136",
                    "hostPath": "/dev/dri/renderD136",
                    "permissions": [
                        "read",
```

```
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD137",
                    "hostPath": "/dev/dri/renderD137",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD138",
                    "hostPath": "/dev/dri/renderD138",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD139",
                    "hostPath": "/dev/dri/renderD139",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD140",
                    "hostPath": "/dev/dri/renderD140",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD141",
                    "hostPath": "/dev/dri/renderD141",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
```

```
                        "containerPath": "/dev/dri/renderD142",
                        "hostPath": "/dev/dri/renderD142",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD143",
                        "hostPath": "/dev/dri/renderD143",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Registrieren Sie die Aufgabendefinition.

```
aws ecs register-task-definition --family vt1-24xlarge-xffmpeg-processor --cli-
input-json file://vt1-24xlarge-xffmpeg-linux.json --region us-east-1
```

Amazon ECS-Aufgabendefinitionen für AWS Neuron-Workloads für maschinelles Lernen

Sie können Instances von Amazon EC2 Trn1, Amazon EC2 Inf1 und Amazon EC2 Inf2 in Ihren Clustern für Machine-Learning-Workloads registrieren.

Amazon-EC2-Trn1-Instances werden von AWS -Trainium-Chips unterstützt. Diese Instances bieten leistungsstarkes und kostengünstiges Training für Machine Learning in der Cloud. Sie können ein Inferenzmodell für Machine Learning mithilfe eines Machine Learning-Frameworks mit AWS Neuron auf einer Trn1-Instance trainieren. Anschließend können Sie das Modell auf einer Inf1-Instance oder einer Inf2-Instance ausführen, um die Beschleunigung der Inferentia-Chips zu nutzen. AWS

Die Amazon-EC2-Instances Inf1 und Inf2 werden von AWS -Inferentia-Chips angetrieben. Sie bieten hohe Leistung und niedrigste Kosten für Inferenzen in der Cloud.

Machine-Learning-Modelle werden auf Containern mit AWS Neuron bereitgestellt, einem spezialisierten Software Developer Kit (SDK). Das SDK besteht aus einem Compiler, Runtime- und Profiling-Tools, die die Leistung von Machine-Learning-Chips für maschinelles Lernen optimieren. AWS AWS Neuron unterstützt beliebte Frameworks für maschinelles Lernen wie TensorFlow PyTorch, und Apache MXNet.

Überlegungen

Berücksichtigen Sie Folgendes, bevor Sie mit der Bereitstellung von Neuron auf Amazon ECS beginnen:

- Ihre Cluster können eine Mischung aus Trn1-, Inf1-, Inf2- und anderen Instances enthalten.
- Sie benötigen eine Linux-Anwendung in einem Container, der ein Framework für maschinelles Lernen verwendet, das Neuron unterstützt AWS .

  > **Important**
  >
  > Anwendungen, die andere Frameworks verwenden, haben möglicherweise keine verbesserte Leistung auf Trn1-, Inf1 und Inf2-Instances.

- Auf jedem AWS Trainium- oder AWS Inferentia-Chip kann nur eine Inferenz- oder Inferenztrainingsaufgabe ausgeführt werden. Für Inf1 hat jeder Chip 4. NeuronCores Für Trn1 und Inf2 hat jeder Chip 2. NeuronCores Sie können so viele Aufgaben ausführen, wie Chips für jede Ihrer Trn1-, Inf1- und Inf2-Instances vorhanden sind.

- Wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen, können Sie beim Konfigurieren der Aufgabenplatzierungbedingungen Attribute des Instance-Typs verwenden. Dadurch wird sichergestellt, dass die Aufgabe auf der von Ihnen angegebenen Container-Instance gestartet wird. Das hilft dabei, die allgemeine Ressourcenauslastung zu optimieren und sicherzustellen, dass sich Aufgaben für Inferenz-Workloads auf Ihren Trn1-, Inf1 oder Inf2-Instances befinden. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

  Im folgenden Beispiel wird eine Aufgabe auf einer `Inf1.xlarge`-Instance auf Ihrem `default`-Cluster ausgeführt.

  ```
  aws ecs run-task \
       --cluster default \
       --task-definition ecs-inference-task-def \
       --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
   Inf1.xlarge"
  ```

- Ressourcenanforderungen für Neuron können in einer Aufgabendefinition nicht definiert werden. Stattdessen konfigurieren Sie einen Container so, dass er bestimmte AWS Trainium- oder AWS Inferentia-Chips verwendet, die auf der Host-Container-Instance verfügbar sind. Nutzen Sie dazu den `linuxParameters`-Parameter und geben Sie die Geräteinformationen an. Weitere Informationen finden Sie unter Anforderungen an die Aufgabendefinition.

Verwenden Sie das Amazon ECS-optimierte Amazon Linux 2023 (Neuron) AMI

Amazon ECS bietet ein für Amazon ECS optimiertes AMI, das auf Amazon Linux 2023 für AWS Trainium- und AWS Inferentia-Workloads basiert. Es wird mit den AWS Neuron-Treibern und der Laufzeit für Docker geliefert. Dieses AMI erleichtert das Ausführen von Machine Learning Inferenz-Workloads auf Amazon ECS.

Wir empfehlen, das Amazon ECS-optimierte Amazon Linux 2023 (Neuron) AMI zu verwenden, wenn Sie Ihre Amazon EC2 Trn1-, Inf1- und Inf2-Instances starten.

Sie können das aktuelle Amazon ECS-optimierte Amazon Linux 2023 (Neuron) AMI AWS CLI mit dem folgenden Befehl abrufen.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/
recommended
```

Das Amazon ECS-optimierte Amazon Linux 2023 (Neuron) AMI wird in den folgenden Regionen unterstützt:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Asien-Pazifik (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Kanada (Zentral)
- Europe (Frankfurt)
- Europa (Irland)
- Europe (London)
- Europe (Paris)
- Europa (Stockholm)
- Südamerika (São Paulo)

Verwenden Sie das für Amazon ECS optimierte Amazon Linux 2 (Neuron) AMI

Amazon ECS bietet ein für Amazon ECS optimiertes AMI, das auf Amazon Linux 2 für AWS Trainium- und AWS Inferentia-Workloads basiert. Es wird mit den AWS Neuron-Treibern und der Laufzeit für Docker geliefert. Dieses AMI erleichtert das Ausführen von Machine Learning Inferenz-Workloads auf Amazon ECS.

Wir empfehlen, das Amazon-ECS-optimierte AMI für Amazon Linux 2 (Neuron) zu verwenden, wenn Sie Ihre Amazon-EC2-Trn1-, -Inf1 und -Inf2-Instances starten.

Sie können das aktuelle Amazon ECS-optimierte Amazon Linux 2 (Neuron) AMI AWS CLI mit dem folgenden Befehl abrufen.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended
```

Das für Amazon ECS optimierte Amazon Linux 2 (Neuron) AMI wird in den folgenden Regionen unterstützt:

- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Asien-Pazifik (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Kanada (Zentral)
- Europe (Frankfurt)
- Europa (Irland)
- Europe (London)
- Europe (Paris)
- Europa (Stockholm)
- Südamerika (São Paulo)

Anforderungen an die Aufgabendefinition

Um Neuron auf Amazon ECS bereitzustellen, muss Ihre Aufgabendefinition die Container-Definition für einen vorgefertigten Container enthalten, für den das Inferenzmodell verwendet wird. TensorFlow Es wird von AWS Deep Learning Containers bereitgestellt. Dieser Container enthält die AWS Neuron-Runtime und die TensorFlow Serving-Anwendung. Beim Start ruft dieser Container Ihr Modell von Amazon S3 ab, startet Neuron TensorFlow Serving mit dem gespeicherten Modell und wartet auf Vorhersageanfragen. Im folgenden Beispiel hat das Container-Image TensorFlow 1.15 und Ubuntu 18.04. Eine vollständige Liste vorgefertigter Deep Learning Containers, die für Neuron optimiert sind,

wird auf geführt. GitHub Weitere Informationen finden Sie unter AWS Neuron TensorFlow Serving verwenden.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-neuron:1.15.4-neuron-py37-ubuntu18.04
```

Alternativ können Sie Ihr eigenes Neuron Seitenwagen Container-Image erstellen. Weitere Informationen finden Sie unter Tutorial: Neuron TensorFlow Serving im AWS Deep Learning AMI Entwicklerhandbuch.

Die Aufgabendefinition muss für den einzelnen Instance-Typ spezifisch sein. Sie müssen einen Container so konfigurieren, dass er bestimmte AWS Trainium- oder AWS Inferentia-Geräte verwendet, die auf der Host-Container-Instance verfügbar sind. Dazu können Sie den `linuxParameters`-Parameter verwenden. In der folgenden Tabelle sind die Chips aufgeführt, die für jeden Instance-Typ spezifisch sind.

Instance-Typ	vCPUs	RAM (GiB)	AWS ML-Beschleunigerchips	Gerätepfade
trn1.2xlarge	8	32	1	`/dev/neuron0`
trn1.32xlarge	128	512	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neur`

Instance-Typ	vCPUs	RAM (GiB)	AWS ML-Beschleunigerchips	Gerätepfade
				`on11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`
inf1.xlarge	4	8	1	`/dev/neuron0`
inf1.2xlarge	8	16	1	`/dev/neuron0`
inf1.6xlarge	24	48	4	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3`

Instance-Typ	vCPUs	RAM (GiB)	AWS ML-Beschleunigerchips	Gerätepfade
inf1.24xlarge	96	192	16	`/dev/neuron0` ,`/dev/neuron1` ,`/dev/neuron2` ,`/dev/neuron3` ,`/dev/neuron4` ,`/dev/neuron5` ,`/dev/neuron6` ,`/dev/neuron7` ,`/dev/neuron8` ,`/dev/neuron9` ,`/dev/neuron10` ,`/dev/neuron11` ,`/dev/neuron12` ,`/dev/neuron13` ,`/dev/neuron14` ,`/dev/neuron15`
inf2.xlarge	8	16	1	`/dev/neuron0`
inf2.8xlarge	32	64	1	`/dev/neuron0`

Instance-Typ	vCPUs	RAM (GiB)	AWS ML-Beschleunigerchips	Gerätepfade
inf2.24xlarge	96	384	6	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,`
inf2.48xlarge	192	768	12	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11`

Spezifizierung des maschinellen Lernens mit AWS Neuron in einer Amazon ECS-Aufgabendefinition

Nachfolgend finden Sie ein Beispiel der Linux-Aufgabendefinition für `inf1.xlarge`, das die zu verwendende Syntax anzeigt.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == inf1.xlarge"
        }
    ],
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "entryPoint": [
                "/usr/local/bin/entrypoint.sh",
                "--port=8500",
                "--rest_api_port=9000",
                "--model_name=resnet50_neuron",
                "--model_base_path=s3://your-bucket-of-models/resnet50_neuron/"
            ],
            "portMappings": [
                {
                    "hostPort": 8500,
                    "protocol": "tcp",
                    "containerPort": 8500
                },
                {
                    "hostPort": 8501,
                    "protocol": "tcp",
                    "containerPort": 8501
                },
                {
                    "hostPort": 0,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "linuxParameters": {
                "devices": [
                    {
```

```
                        "containerPath": "/dev/neuron0",
                        "hostPath": "/dev/neuron0",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ],
                "capabilities": {
                    "add": [
                        "IPC_LOCK"
                    ]
                }
            },
            "cpu": 0,
            "memoryReservation": 1000,
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-
inference-neuron:1.15.4-neuron-py37-ubuntu18.04",
            "essential": true,
            "name": "resnet50"
        }
    ]
}
```

Amazon ECS-Aufgabendefinitionen für Deep-Learning-Instances

Um Deep-Learning-Workloads auf Amazon ECS zu verwenden, registrieren Sie Amazon-EC2-DL1-Instances für Ihre Cluster. Amazon-EC2-DL1-Instances basieren auf Gaudi-Accelerators von Habana Labs (einem Intel-Unternehmen). Verwenden Sie das Habana SynapSeai SDK, um eine Verbindung zu den Habana Gaudi-Accelerators herzustellen. Das SDK unterstützt die beliebten Frameworks für maschinelles Lernen und TensorFlow . PyTorch

Überlegungen

Seien Sie sich der folgenden Überlegungen bewusst, bevor Sie mit der Bereitstellung von DL1 auf Amazon ECS beginnen:

- Ihre Cluster können eine Mischung aus DL1- und Nicht-DL1-Instances enthalten.
- Wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen, können Sie insbesondere beim Konfigurieren der Aufgabenplatzierungsbedingungen sicherstellen, dass Ihre Aufgabe auf der von Ihnen angegebenen Container-Instance gestartet wird. Dadurch wird

sichergestellt, dass Ihre Ressourcen effektiv eingesetzt werden und dass sich Ihre Aufgaben für Deep-Learning-Workloads auf Ihren DL1-Instances befinden. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

Im folgenden Beispiel wird eine Aufgabe für eine `dl1.24xlarge`-Instance auf Ihrem `default`-Cluster ausgeführt.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-dl1-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 dl1.24xlarge"
```

Verwenden eines DL1-AMI

Sie haben drei Möglichkeiten, ein AMI auf Amazon-EC2-DL1-Instances für Amazon ECS auszuführen:

- AWS Marketplace AMIs, die von Habana hier bereitgestellt werden.
- Habana Deep Learning AMIs, die von Amazon Web Services bereitgestellt werden. Weil er nicht enthalten ist, müssen Sie den Amazon-ECS-Container-Agent separat installieren.
- Verwenden Sie Packer, um ein benutzerdefiniertes AMI zu erstellen, das vom GitHubRepo bereitgestellt wird. Weitere Informationen finden Sie in der Packer-Dokumentation.

Spezifizierung von Deep Learning in einer Amazon ECS-Aufgabendefinition

Um beschleunigte Deep Learning-Container von Habana Gaudi auf Amazon ECS auszuführen, muss Ihre Aufgabendefinition die Containerdefinition für einen vorgefertigten Container enthalten, der das Deep-Learning-Modell für TensorFlow oder PyTorch mithilfe von Habana SynapseAI bedient, das von Deep Learning Containers bereitgestellt wird. AWS

Das folgende Container-Image hat 2.7.0 und Ubuntu 20.04. TensorFlow Eine vollständige Liste vorgefertigter Deep Learning Containers, die für die Gaudi-Beschleuniger von Habana optimiert sind, wird auf geführt. GitHub Weitere Informationen finden Sie unter Habana Training Containers.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training-habana:2.7.0-hpu-py38-
synapseai1.2.0-ubuntu20.04
```

Nachfolgend finden Sie ein Beispiel für eine Aufgabendefinition für Linux-Container auf Amazon EC2, das die zu verwendende Syntax veranschaulicht. In diesem Beispiel wird ein Image verwendet, das das Habana Labs System Management Interface Tool (HL-SMI) enthält, das hier zu finden ist: `vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/tensorflow-installer-tf-cpu-2.6.0:1.1.0-614`

```
{
    "family": "dl-test",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == dl1.24xlarge"
        }
    ],
    "networkMode": "host",
    "cpu": "10240",
    "memory": "1024",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": ["hl-smi"],
            "cpu": 8192,
            "environment": [
                {
                    "name": "HABANA_VISIBLE_DEVICES",
                    "value": "all"
                }
            ],
            "image": "vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/
tensorflow-installer-tf-cpu-2.6.0:1.1.0-614",
            "essential": true,
            "name": "tensorflow-installer-tf-hpu"
        }
    ]
```

```
}
```

Amazon ECS-Aufgabendefinitionen für 64-Bit-ARM-Workloads

Amazon ECS unterstützt die Verwendung von 64-Bit-ARM-Anwendungen. Sie können Ihre Anwendungen auf der Plattform ausführen, die von Graviton2-Prozessoren angetrieben wird.AWS Sie ist für eine Vielzahl von Workloads geeignet. Dazu gehören Workloads wie Anwendungsserver, Microservices, Hochleistungs-Computing, CPU-basierte Machine Learning-Inferenz, Videocodierung, elektronische Designautomatisierung, Spiele, Open-Source-Datenbanken und In-Memory-Caches.

Überlegungen

Beachten Sie die Folgendes, bevor Sie damit beginnen, Aufgabendefinitionen bereitzustellen, die die 64-Bit-ARM-Architektur verwenden:

- Die Anwendungen können die Launchtypen Fargate oder EC2 verwenden.
- Linux-Aufgaben mit der ARM64-Architektur unterstützen den Fargate-Spot-Kapazitätsanbieter nicht.
- Die Anwendungen können nur das Linux-Betriebssystem verwenden.
- Für den Fargate-Typ müssen die Anwendungen die Fargate-Plattformversion `1.4.0` oder höher verwenden.
- Die Anwendungen können Fluent Bit oder CloudWatch zur Überwachung verwenden.
- Für den Starttyp Fargate unterstützen die folgenden Programme AWS-Regionen keine 64-Bit-ARM-Workloads:
  - USA Ost (Nord-Virginia), die `use1-az3` Availability Zone
- Für den Amazon-EC2-Starttyp, beachten Sie Folgendes, um zu überprüfen, ob die Region, in der Sie sich befinden, den gewünschten Instance-Typ unterstützt:
  - Amazon-EC2-M6g-Instances
  - Amazon-EC2-T4g-Instances
  - Amazon-EC2-C6g-Instances
  - Amazon-EC2-R6gd-Instances
  - Amazon-EC2-X2gd-Instances

  Sie können auch den Amazon-EC2-Befehl `describe-instance-type-offerings` mit einem Filter verwenden, um das Instance-Angebot für Ihre Region anzuzeigen.

```
aws ec2 describe-instance-type-offerings --filters Name=instance-
type,Values=instance-type --region region
```

Im folgenden Beispiel wird die Verfügbarkeit des M6-Instance-Typs in der Region USA Ost (Nord-Virginia) (us-east-1) überprüft.

```
aws ec2 describe-instance-type-offerings --filters "Name=instance-type,Values=m6*" --
region us-east-1
```

Weitere Informationen finden Sie describe-instance-type-offerings in der Amazon EC2 EC2-Befehlszeilenreferenz.

Spezifizierung der ARM-Architektur in einer Amazon ECS-Aufgabendefinition

Um die ARM-Architektur zu verwenden, geben Sie `ARM64` für den Aufgabendefinitions-Parameter `cpuArchitecture` an.

Im folgenden Beispiel wird die ARM-Architektur in einer Aufgabendefinition angegeben. Sie ist im JSON-Format.

```
{
    "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
    },
...
}
```

Im folgenden Beispiel zeigt eine Aufgabendefinition für die ARM-Architektur „Hallo Welt“ an.

```
{
 "family": "arm64-testapp",
 "networkMode": "awsvpc",
 "containerDefinitions": [
    {
        "name": "arm-container",
        "image": "arm64v8/busybox",
        "cpu": 100,
        "memory": 100,
```

```
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
 ],
 "requiresCompatibilities": [ "FARGATE" ],
 "cpu": "256",
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Amazon ECS-Protokolle senden an CloudWatch

Sie können die Container in Ihren Aufgaben so konfigurieren, dass Protokollinformationen an CloudWatch Logs gesendet werden. Wenn Sie den Starttyp Fargate für Ihre Aufgaben verwenden, können Sie die Protokolle von Ihren Container einsehen. Wenn Sie den EC2-Starttyp verwenden, können Sie verschiedene Protokolle Ihrer Container bequem an einem Ort aufrufen und es wird verhindert, dass Ihre Containerprotokolle Speicherplatz in Ihren Container-Instances belegen.

> **Note**
>
> Der Typ der Informationen, die von den Containern in Ihrer Aufgabe protokolliert werden, hängt sehr stark von ihrem `ENTRYPOINT`-Befehl ab. Standardmäßig zeigen die erfassten Protokolle die Befehlsausgabe an, die Sie normalerweise in einem interaktiven Terminal sehen würden, wenn Sie den Container lokal ausführen. Dabei handelt es sich um die `STDOUT`- und `STDERR`-E/A-Streams. Der `awslogs` Protokolltreiber leitet diese Protokolle einfach von Docker an CloudWatch Logs weiter. Weitere Informationen dazu, wie Docker-Protokolle verarbeitet werden, einschließlich alternativer Möglichkeiten zum Erfassen unterschiedlicher Dateidaten oder Streams, finden Sie unter Anzeigen von Protokollen für einen Container oder Service in der Docker-Dokumentation.

Informationen zum Senden von Systemprotokollen von Ihren Amazon ECS-Container-Instances an CloudWatch Logs finden Sie unter Überwachung von Protokolldateien und CloudWatch Protokollkontingenten im Amazon CloudWatch Logs-Benutzerhandbuch.

Fargate Starttyp

Wenn Sie den Starttyp Fargate für Ihre Aufgaben verwenden, müssen Sie für die Aktivierung des `logConfiguration`-Protokolltreibers nur die erforderlichen `awslogs`-Parameter in Ihre Aufgabendefinition einfügen. Weitere Informationen finden Sie unter Beispiel für eine Amazon ECS-Aufgabendefinition: Protokolle weiterleiten an CloudWatch.

Führen Sie für Windows-Container auf Fargate eine der folgenden Optionen aus, wenn einer Ihrer Aufgabendefinitionsparameter Sonderzeichen wie, & `\` `<` `>` `^` `|` enthält:

- Fügen Sie ein Escapezeichen (\) mit doppelten Anführungszeichen um die gesamte Parameterzeichenfolge

  Beispiel

  ```
  "awslogs-multiline-pattern": "\"^[|DEBUG|INFO|WARNING|ERROR\"",
  ```

- Fügen Sie jedem Sonderzeichen ein Escape-Zeichen (^) hinzu

  Beispiel

  ```
  "awslogs-multiline-pattern": "^^[^|DEBUG^|INFO^|WARNING^|ERROR",
  ```

EC2-Starttyp

Wenn Sie den Starttyp EC2 für Ihre Aufgaben verwenden und den Protokolltreiber `awslogs` aktivieren möchten, benötigen Ihre Amazon ECS Container-Instances mindestens Version 1.9.0 des Container-Agenten. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

> **Note**
>
> Sie müssen entweder ein Amazon ECS-optimiertes AMI oder ein benutzerdefiniertes AMI mit mindestens einer Version `1.9.0-1` des `ecs-init` Pakets verwenden. Wenn Sie ein benutzerdefiniertes AMI verwenden, müssen Sie angeben, dass der `awslogs` Protokollierungstreiber auf der Amazon EC2 EC2-Instance verfügbar ist, wenn Sie den Agenten starten, indem Sie die folgende Umgebungsvariable in Ihrer docker run Anweisung oder Umgebungsvariablendatei verwenden.

```
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
```

Ihre Amazon-ECS-Container-Instances benötigen auch eine `logs:CreateLogStream`- und `logs:PutLogEvents`-Berechtigung für die IAM-Rolle, mit der Sie Ihre Container-Instances starten. Wenn Sie die Amazon-ECS-Container-Instance-Rolle erstellt haben, bevor die Unterstützung des Protokolltreibers `awslogs` in Amazon ECS aktiviert war, müssen Sie diese Berechtigung eventuell hinzufügen. Die `ecsTaskExecutionRole` wird verwendet, wenn sie der Aufgabe zugewiesen ist und wahrscheinlich die richtigen Berechtigungen enthält. Informationen zur Rolle „Aufgabenausführung“ finden Sie unterIAM-Rolle für die Amazon-ECS-Aufgabenausführung. Wenn Ihre Container-Instances die verwaltete IAM-Richtlinie für Container-Instances verwenden, verfügen sie wahrscheinlich über die korrekten Berechtigungen. Informationen zur verwalteten IAM-Richtlinie für Container-Instances finden Sie unterIAM-Rolle für Amazon-ECS-Container-Instance.

Beispiel für eine Amazon ECS-Aufgabendefinition: Protokolle weiterleiten an CloudWatch

Bevor Ihre Container Protokolle an senden können CloudWatch, müssen Sie den `awslogs` Protokolltreiber für Container in Ihrer Aufgabendefinition angeben. Weitere Hinweise zu den Protokollparametern finden Sie unter Speicher und Protokollierung

Bei der unten gezeigten Aufgabendefinition JSON wurde ein `logConfiguration`-Objekt für jeden Container festgelegt. Einer davon ist für den WordPress Container, der Protokolle an eine Protokollgruppe namens sendet`awslogs-wordpress`. Das andere für einen MySQL-Container, der Protokolle an eine Protokollgruppe mit dem Namen `awslogs-mysql` sendet. Beide Container verwenden den Protokoll-Stream-Präfix `awslogs-example`.

```
{
    "containerDefinitions": [
        {
            "name": "wordpress",
            "links": [
                "mysql"
            ],
            "image": "wordpress",
            "essential": true,
            "portMappings": [
                {
```

```
                "containerPort": 80,
                "hostPort": 80
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-create-group": "true",
                "awslogs-group": "awslogs-wordpress",
                "awslogs-region": "us-west-2",
                "awslogs-stream-prefix": "awslogs-example"
            }
        },
        "memory": 500,
        "cpu": 10
    },
    {
        "environment": [
            {
                "name": "MYSQL_ROOT_PASSWORD",
                "value": "password"
            }
        ],
        "name": "mysql",
        "image": "mysql",
        "cpu": 10,
        "memory": 500,
        "essential": true,
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-create-group": "true",
                "awslogs-group": "awslogs-mysql",
                "awslogs-region": "us-west-2",
                "awslogs-stream-prefix": "awslogs-example",
                "mode": "non-blocking",
                "max-buffer-size": "25m"
            }
        }
    }
],
"family": "awslogs-example"
}
```

Nachdem Sie eine Aufgabendefinition beim `awslogs` Protokolltreiber in einer Protokollkonfiguration für die Containerdefinition registriert haben, können Sie eine Aufgabe ausführen oder einen Dienst mit dieser Aufgabendefinition erstellen, um mit dem Senden von Protokollen an Logs zu CloudWatch beginnen. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen und Einen Amazon ECS-Service mithilfe der Konsole erstellen.

Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner

Sie können Amazon ECS verwenden FireLens , um Aufgabendefinitionsparameter zu verwenden, um Protokolle zur Protokollspeicherung und Analyse an ein AWS Service- oder AWS Partner Network (APN-) Ziel weiterzuleiten. The AWS Partner Network ist eine globale Partnergemeinschaft, die Programme, Fachwissen und Ressourcen nutzt, um Kundenangebote aufzubauen, zu vermarkten und zu verkaufen. Weitere Informationen finden Sie unter AWS Partner. FireLens arbeitet mit Fluentd und Fluent Bit. Wir stellen AWS für das Fluent Bit-Image bereit oder Sie können Ihr eigenes Fluentd- oder Fluent Bit-Image verwenden.

Beachten Sie bei der Verwendung FireLens für Amazon ECS Folgendes:

- Wir empfehlen, dass Sie `my_service_` dem Protokoll einen Container-Namen hinzufügen, damit Sie die Container-Namen in der Konsole leicht unterscheiden können.
- Amazon ECS fügt standardmäßig eine Abhängigkeit von der Reihenfolge der Startcontainer zwischen den Anwendungscontainern und dem FireLens Container hinzu. Wenn Sie eine Container-Reihenfolge zwischen den Anwendungscontainern und dem FireLens Container angeben, wird die standardmäßige Reihenfolge der Startcontainer außer Kraft gesetzt.
- FireLensfür Amazon ECS wird für Aufgaben unterstützt, die sowohl auf Linux als auch AWS Fargate auf Amazon EC2 unter Linux gehostet werden. Windows-Container unterstützen FireLens nicht.

  Informationen zur Konfiguration der zentralen Protokollierung für Windows-Container finden Sie unter Zentralisierte Protokollierung für Windows-Container auf Amazon ECS mit Fluent Bit.
- Sie können AWS CloudFormation Vorlagen verwenden, um FireLens für Amazon ECS zu konfigurieren. Weitere Informationen finden Sie AWS::ECS::TaskDefinition FirelensConfigurationim Benutzerhandbuch AWS CloudFormation
- FireLensüberwacht den Port. Um also sicherzustellen`24224`, dass der FireLens Log Router außerhalb der Aufgabe nicht erreichbar ist, dürfen Sie keinen eingehenden Verkehr über den Port `24224` in der Sicherheitsgruppe zulassen, die Ihre Aufgabe verwendet. Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, ist dies die der Aufgabe zugeordnete Sicherheitsgruppe.

Für Aufgaben, die den `host`-Netzwerkmodus verwenden, ist dies die Sicherheitsgruppe, die der Amazon-EC2-Instance zugeordnet ist, die die Aufgabe hostet. Für Aufgaben, die den `bridge`-Netzwerkmodus verwenden, erstellen Sie keine Portzuordnungen, die Port `24224` verwenden.

- Bei Aufgaben, die den `bridge` Netzwerkmodus verwenden, muss der Container mit der FireLens Konfiguration vor allen Anwendungscontainern, die darauf angewiesen sind, gestartet werden. Um die Startreihenfolge Ihrer Container zu steuern, verwenden Sie Abhängigkeitsbedingungen in der Aufgabendefinition. Weitere Informationen finden Sie unter Container-Abhängigkeit.

  > **Note**
  >
  > Wenn Sie Parameter für Abhängigkeitsbedingungen in Containerdefinitionen mit einer FireLens Konfiguration verwenden, stellen Sie sicher, dass für jeden Container eine `START` `HEALTHY` Oder-Bedingung erforderlich ist.

- Standardmäßig fügt FireLens den Cluster- und Aufgabendefinitionsnamen und den Amazon-Ressourcennamen (ARN) des Clusters als Metadatenschlüssel zu Ihren stdout/stderr-Containerprotokollen hinzu. Nachfolgend ist ein Beispiel für das Metadatenformat.

  ```
  "ecs_cluster": "cluster-name",
  "ecs_task_arn": "arn:aws:ecs:region:111122223333:task/cluster-
  name/f2ad7dba413f45ddb4EXAMPLE",
  "ecs_task_definition": "task-def-name:revision",
  ```

  Wenn Sie die Metadaten nicht in Ihren Protokollen haben möchten, setzen Sie `enable-ecs-log-metadata` zu `false` im `firelensConfiguration`-Abschnitt der Aufgabendefinition.

  ```
  "firelensConfiguration":{
     "type":"fluentbit",
     "options":{
        "enable-ecs-log-metadata":"false",
        "config-file-type":"file",
        "config-file-value":"/extra.conf"
  }
  ```

Um diese Funktion verwenden zu können, müssen Sie eine IAM-Rolle für Ihre Aufgaben erstellen, die für die Nutzung aller AWS Dienste erforderlich sind, die für die Aufgaben erforderlich sind. Wenn ein Container beispielsweise Logs an Firehose weiterleitet, benötigt die Aufgabe die Erlaubnis, die

`firehose:PutRecordBatch` API aufzurufen. Informationen finden Sie im Abschnitt Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

Unter den folgenden Bedingungen ist für Ihre Aufgabe möglicherweise auch die Amazon ECS-Aufgabenausführungsrolle erforderlich. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

- Wenn Ihre Aufgabe auf Fargate gehostet wird und Sie Container-Images aus Amazon ECR abrufen oder sensible Daten aus AWS Secrets Manager Ihrer Protokollkonfiguration referenzieren, müssen Sie die IAM-Rolle für die Aufgabenausführung angeben.
- Wenn Sie eine benutzerdefinierte Konfigurationsdatei verwenden, die in Amazon S3 gehostet wird, muss Ihre IAM-Rolle für die Aufgabenausführung die `s3:GetObject` entsprechende Berechtigung enthalten.

Informationen zur Verwendung mehrerer Konfigurationsdateien mit Amazon ECS, einschließlich Dateien, die Sie hosten, oder Dateien in Amazon S3, finden Sie unter Init-Prozess für Fluent Bit auf ECS, Multi-Config-Unterstützung.

Konfiguration von Amazon ECS-Protokollen für hohen Durchsatz

Wenn Sie eine Aufgabendefinition erstellen, können Sie die Anzahl der Protokollzeilen angeben, die im Speicher gepuffert werden, indem Sie den Wert in der angeben. `log-driver-buffer-limit` Weitere Informationen finden Sie unter Fluentd-Protokollierungstreiber in der Docker-Dokumentation.

Verwenden Sie diese Option bei hohem Durchsatz, da Docker möglicherweise nicht mehr genügend Pufferspeicher hat und Puffernachrichten verwirft, sodass neue Nachrichten hinzugefügt werden können.

Beachten Sie bei der Verwendung FireLens für Amazon ECS mit der Option Buffer Limit Folgendes:

- Diese Option wird vom Amazon-EC2-Launchtyp und vom Fargate-Launchtyp mit Plattformversion `1.4.0` oder höher unterstützt.
- Die Option ist nur gültig, wenn `logDriver` auf `awsfirelens` gesetzt ist.
- Das Standard-Pufferlimit ist `1048576` Protokollzeilen.
- Die gültigen Werte sind `0` und `536870912` Log-Zeilen.
- Die maximale Speichermenge, die für diesen Puffer verwendet wird, ist das Produkt aus der Größe jeder Protokollzeile und der Größe des Puffers. Wenn die Protokollzeilen der Anwendung

beispielsweise im Durchschnitt 2 KiB betragen, würde ein Pufferlimit von 4096 höchstens 8 MiB verbrauchen. Die Gesamtmenge des auf Aufgabenebene zugewiesenen Speichers sollte größer sein als die Speichermenge, die allen Containern zusätzlich zum Speicherpuffer des Protokolltreibers zugewiesen ist.

Wenn der `awsfirelens`-Protokolltreiber in einer Aufgabendefinition angegeben wird, injiziert der Amazon ECS Container-Agent die folgenden Umgebungsvariablen in den Container:

`FLUENT_HOST`

Die IP-Adresse, die dem FireLens Container zugewiesen ist.

`FLUENT_PORT`

Der Port, über den das Fluent Forward-Protokoll kommuniziert.

Sie können die Umgebungsvariablen `FLUENT_HOST` und `FLUENT_PORT` verwenden, um Protokolle direkt vom Code aus in den Protokoll-Router zu schreiben, anstatt durch `stdout` zu gehen. Weitere Informationen finden Sie unter fluent-logger-golang on. GitHub

Im Folgenden wird die Syntax für die Angabe von gezeigt. `log-driver-buffer-limit` `my_service_`Ersetzen Sie es durch den Namen Ihres Dienstes:

```
{
    "containerDefinitions": [
        {
            "essential": true,
            "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
            "name": "my_service_log_router",
            "firelensConfiguration": {
                "type": "fluentbit"
            },
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
```

```
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "httpd",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
                    "log-driver-buffer-limit": "51200"
                }
            },
            "dependsOn": [
                {
                    "containerName": "log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

AWS für Fluent Bit Image-Repositorys für Amazon ECS

AWS bietet ein Fluent Bit Image mit Plugins für CloudWatch Logs und Firehose. Es wird empfohlen, Fluent Bit als Protokoll-Router zu verwenden, da es eine geringere Ressourcenauslastung aufweist als Fluentd. Weitere Informationen finden Sie unter CloudWatch Logs for Fluent Bit und Amazon Kinesis Firehose for Fluent Bit.

Das AWS for Fluent Bit-Bild ist auf Amazon ECR sowohl in der Amazon ECR Public Gallery als auch in einem Amazon ECR-Repository in den meisten AWS-Regionen Fällen für hohe Verfügbarkeit verfügbar.

Amazon ECR Public Gallery

Das Fluent Bit Bild AWS für ist in der Amazon ECR Public Gallery verfügbar. Dies ist der empfohlene Ort, um das Fluent Bit Bild AWS für herunterzuladen, da es sich um ein öffentliches Repository

handelt, das von allen AWS-Regionen genutzt werden kann. Weitere Informationen finden Sie unter aws-for-fluent-bit auf der Amazon ECR Public Gallery.

Linux

Das Fluent Bit Bild AWS for in der Amazon ECR Public Gallery unterstützt das Amazon Linux-Betriebssystem mit der `ARM 64` oder `x86-64` -Architektur.

Sie können das AWS Fluent Bit For-Bild aus der Amazon ECR Public Gallery abrufen, indem Sie die Repository-URL mit dem gewünschten Image-Tag angeben. Die verfügbaren Image-Tags finden Sie in der Registerkarte Image-Tags auf der Amazon ECR Public Gallery.

Nachfolgend finden Sie die für die Docker-CLI zu verwendende Syntax.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Mit diesem Docker-CLI-Befehl können Sie beispielsweise das neueste stabile AWS Fluent Bit For-Image abrufen.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:stable
```

> **Note**
>
> Nicht authentifizierte Pulls sind zulässig, haben jedoch eine niedrigere Rate als authentifizierte Pulls. Verwenden Sie den folgenden Befehl, um sich vor dem Abrufen mit Ihrem AWS Konto zu authentifizieren.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
>  AWS --password-stdin public.ecr.aws
> ```

Windows

Das Fluent Bit Bild AWS for in der Amazon ECR Public Gallery unterstützt die `AMD64` Architektur mit den folgenden Betriebssystemen:

- Windows Server 2022 Voll
- Windows Server 2022 Kern
- Windows Server 2019 Voll

- Windows Server 2019 Kern

Windows-Container, die sich auf AWS Fargate befinden, werden nicht unterstützt FireLens.

Sie können das AWS Fluent Bit For-Bild aus der Amazon ECR Public Gallery abrufen, indem Sie die Repository-URL mit dem gewünschten Image-Tag angeben. Die verfügbaren Image-Tags finden Sie in der Registerkarte Image-Tags auf der Amazon ECR Public Gallery.

Nachfolgend finden Sie die für die Docker-CLI zu verwendende Syntax.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Mit diesem Docker-CLI-Befehl können Sie beispielsweise den neuesten Stable AWS for Fluent Bit Image abrufen.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:windowsservercore-
stable
```

> **Note**
>
> Nicht authentifizierte Pulls sind zulässig, haben jedoch eine niedrigere Rate als authentifizierte Pulls. Verwenden Sie den folgenden Befehl, um sich vor dem Abrufen mit Ihrem AWS Konto zu authentifizieren.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
>  AWS --password-stdin public.ecr.aws
> ```

Amazon ECR

Das AWS for Fluent Bit-Bild ist auf Amazon ECR für hohe Verfügbarkeit verfügbar. Diese Bilder sind in den meisten Versionen verfügbar AWS-Regionen, darunter. AWS GovCloud (US)

Linux

Die neueste stabile Bild-URI AWS für Fluent Bit kann mit dem folgenden Befehl abgerufen werden.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/stable \
```

```
      --region us-east-1
```

Alle Versionen des AWS for Fluent Bit-Images können mit dem folgenden Befehl aufgelistet werden, um den Systems Manager Parameter Store-Parameter abzufragen.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit \
      --region us-east-1
```

Das neueste stabile Bild AWS für Fluent Bit kann in einer AWS CloudFormation Vorlage referenziert werden, indem auf den Namen des Systems Manager Manager-Parameterspeichers verwiesen wird. Im Folgenden wird ein Beispiel gezeigt:

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/stable
```

Windows

Die neueste stabile Bild-URI AWS für Fluent Bit kann mit dem folgenden Befehl abgerufen werden.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/windowsservercore-stable \
      --region us-east-1
```

Alle Versionen des AWS for Fluent Bit-Images können mit dem folgenden Befehl aufgelistet werden, um den Systems Manager Parameter Store-Parameter abzufragen.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit/windowsservercore \
      --region us-east-1
```

Das neueste stabile Bild AWS für Fluent Bit kann in einer AWS CloudFormation Vorlage referenziert werden, indem auf den Namen des Systems Manager Manager-Parameterspeichers verwiesen wird. Im Folgenden wird ein Beispiel gezeigt:

```
Parameters:
```

```
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/windowsservercore-stable
```

Beispiel für eine Amazon ECS-Aufgabendefinition: Protokolle weiterleiten an FireLens

Um ein benutzerdefiniertes Protokoll-Routing mit FireLens zu verwenden, müssen Sie Folgendes in der Aufgabendefinition angeben:

- Ein Protokoll-Router-Container, der eine FireLens-Konfiguration enthält. Wir empfehlen, dass der Container als `essential` markiert wird.
- Ein oder mehrere Anwendungscontainer, die eine Protokollkonfiguration enthalten, die den `awsfirelens`-Protokolltreiber angibt.
- Ein Amazon-Ressourcenname (ARN) der Aufgaben-IAM-Rolle, der die Berechtigungen enthält, die für die Aufgabe zum Weiterleiten der Protokolle erforderlich sind.

Beim Erstellen einer neuen Aufgabendefinition mithilfe von gibt es einen FireLens Integrationsbereich, der das Hinzufügen eines Protokollrouter-Containers erleichtert. AWS Management Console Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Amazon ECS konvertiert die Protokollkonfiguration und generiert die Fluentd- oder Fluent Bit-Ausgabekonfiguration. Die Ausgabekonfiguration ist in dem Protokoll-Routing-Container unter `/fluent-bit/etc/fluent-bit.conf` für Fluent Bit und `/fluentd/etc/fluent.conf` für Fluentd eingebaut.

> **Important**
>
> FireLens überwacht Port `24224`. Um sicherzustellen, dass der FireLens Log Router außerhalb der Aufgabe nicht erreichbar ist, dürfen Sie daher keinen eingehenden Datenverkehr über den Port `24224` in der Sicherheitsgruppe zulassen, die Ihre Aufgabe verwendet. Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, ist dies die der Aufgabe zugeordnete Sicherheitsgruppe. Für Aufgaben, die den `host`-Netzwerkmodus verwenden, ist dies die Sicherheitsgruppe, die der Amazon-EC2-Instance zugeordnet ist, die die Aufgabe hostet. Für Aufgaben, die den `bridge`-Netzwerkmodus verwenden, erstellen Sie keine Portzuordnungen, die Port `24224` verwenden.

In den Protokolleinträgen werden standardmäßig zusätzliche Felder von Amazon ECS hinzugefügt, mit denen die Quelle der Protokolle identifiziert werden kann.

- `ecs_cluster` – Der Name des Clusters, zu dem die Aufgabe gehört.
- `ecs_task_arn` – Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört.
- `ecs_task_definition` – Der Name und die Revision der Aufgabendefinition, die die Aufgabe verwendet.
- `ec2_instance_id` – Die Amazon-EC2-Instance-ID, auf der der Container gehostet wird. Dieses Feld ist nur für Aufgaben mit dem Starttyp EC2 gültig.

Sie können den Wert `enable-ecs-log-metadata` auf setzen, `false` wenn Sie die Metadaten nicht benötigen.

Das folgende Beispiel für eine Aufgabendefinition definiert einen Protokoll-Router-Container, der Fluent Bit verwendet, um seine Protokolle an Logs CloudWatch weiterzuleiten. Es definiert auch einen Anwendungscontainer, der eine Protokollkonfiguration verwendet, um Protokolle an Amazon Data Firehose weiterzuleiten, und legt den Speicher, der zum Puffern von Ereignissen verwendet wird, auf 2 MiB fest.

> **Note**
>
> Weitere Beispielaufgabendefinitionen finden Sie unter Amazon FireLens ECS-Beispiele unter GitHub.

```
{
  "family": "firelens-example-firehose",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
```

```
          "enable-ecs-log-metadata": "true"
        }
      },
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "firelens-container",
          "awslogs-region": "us-west-2",
          "awslogs-create-group": "true",
          "awslogs-stream-prefix": "firelens"
        }
      },
      "memoryReservation": 50
    },
    {
      "essential": true,
      "image": "httpd",
      "name": "app",
      "logConfiguration": {
        "logDriver": "awsfirelens",
        "options": {
          "Name": "firehose",
          "region": "us-west-2",
          "delivery_stream": "my-stream",
          "log-driver-buffer-limit": "2097152"
        }
      },
      "memoryReservation": 100
    }
  ]
}
```

Die Schlüsselwertpaare, die als Optionen im `logConfiguration`-Objekt angegeben werden, werden verwendet, um die Fluentd- oder Fluent Bit-Ausgabekonfiguration zu generieren. Im Folgenden finden Sie ein Codebeispiel aus einer Fluent Bit-Ausgabedefinition.

```
[OUTPUT]
    Name   firehose
    Match  app-firelens*
    region us-west-2
    delivery_stream my-stream
```

> **Note**
>
> FireLens verwaltet die `match` Konfiguration. Sie geben die `match` Konfiguration nicht in Ihrer Aufgabendefinition an.

Verwenden Sie eine benutzerdefinierte Konfigurationsdatei

Sie können eine benutzerdefinierte Konfigurationsdatei angeben. Das Konfigurationsdateiformat ist das native Format für den verwendeten Protokoll-Router. Weitere Informationen finden Sie unter Fluentd-Konfigurationsdateisyntax und Fluent-Bit-Konfigurationsdatei.

In Ihrer benutzerdefinierten Konfigurationsdatei sollten Sie für Aufgaben, die den `bridge`- oder `awsvpc`-Netzwerkmodus verwenden, keine Fluentd- oder Fluent-Bit-Weiterleitungseingabe über TCP festlegen, da FireLens diese zur Eingabekonfiguration hinzufügt.

Ihre FireLens-Konfiguration muss die folgenden Optionen enthalten, um eine benutzerdefinierte Konfigurationsdatei anzugeben:

`config-file-type`

Den Quellspeicherort der benutzerdefinierten Konfigurationsdatei. Die verfügbaren Optionen sind `s3` oder `file`.

> **Note**
>
> Aufgaben, die auf gehostet werden, unterstützen AWS Fargate nur den `file` Konfigurationsdateityp.

`config-file-value`

Die Quelle für die benutzerdefinierte Konfigurationsdatei. Wenn der `s3`-Konfigurationsdateityp verwendet wird, ist der Wert der Konfigurationsdatei der vollständige ARN des Amazon S3-Buckets und der Datei. Wenn der `file`-Konfigurationsdateityp verwendet wird, ist der Wert der Konfigurationsdatei der vollständige Pfad der Konfigurationsdatei, die entweder im Container-Image oder auf einem Volume vorhanden ist, das im Container bereitgestellt wird.

> **Important**
>
> Wenn Sie eine benutzerdefinierte Konfigurationsdatei verwenden, müssen Sie einen anderen Pfad als den von FireLens verwendeten angeben. Amazon ECS behält den `/fluent-bit/etc/fluent-bit.conf`-Dateipfad für Fluent Bit und `/fluentd/etc/fluent.conf` für Fluentd.

Das folgende Beispiel zeigt die Syntax, die erforderlich ist, wenn eine benutzerdefinierte Konfiguration angegeben wird.

> **Important**
>
> Um eine benutzerdefinierte Konfigurationsdatei anzugeben, die in Amazon S3 gehostet wird, stellen Sie sicher, dass Sie eine IAM-Aufgabenausführungsrolle mit den entsprechenden Berechtigungen erstellt haben.

Im Folgenden wird die Syntax gezeigt, die für die Angabe einer benutzerdefinierten Konfiguration erforderlich ist.

```
{
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "config-file-type": "s3 | file",
          "config-file-value": "arn:aws:s3:::mybucket/fluent.conf | filepath"
        }
      }
    }
  ]
}
```

> **Note**
>
> Aufgaben, die auf gehostet werden, unterstützen AWS Fargate nur den `file` Konfigurationsdateityp.

Verwenden von AWS Nicht-Container-Images in Amazon ECS

Verwenden Sie die private Registrierung, um Ihre Anmeldeinformationen darin zu speichern AWS Secrets Manager, und verweisen Sie dann in Ihrer Aufgabendefinition darauf. Auf diese Weise können Sie in Ihren Aufgabendefinitionen auf Container-Images verweisen, die in privaten Registern existieren und für AWS die eine Authentifizierung erforderlich ist. Dieses Feature wird durch Aufgaben unterstützt, die auf Fargate, Amazon-EC2-Instances und externen Instances mit Amazon ECS Anywhere gehostet werden.

> **Important**
>
> Wenn Ihre Aufgabendefinition auf ein im Amazon ECR gespeichertes Image verweist, trifft dieses Thema nicht zu. Weitere Informationen finden Sie unter Amazon ECR-Images mit Amazon ECS im Amazon Elastic Container-Registry-Benutzerhandbuch.

Für Aufgaben, die auf Amazon-EC2-Instances gehostet werden, erfordert dieses Feature Version `1.19.0` oder höher des Container-Agenten. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Für Aufgaben, die auf Fargate gehostet werden, erfordert dieses Feature die Plattformversion `1.2.0` oder höher. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

Geben Sie in Ihrer Containerdefinition das Objekt `repositoryCredentials` mit den Details des von Ihnen erstellten Geheimnisses an. Der geheime Schlüssel, auf den verwiesen wird, kann von einem anderen AWS-Region oder einem anderen Konto als der Aufgabe stammen, die ihn verwendet.

> **Note**
>
> Wenn Sie die Amazon ECS-API oder das AWS SDK verwenden und das Geheimnis in derselben AWS-Region Aufgabe enthalten ist, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Geheimnisses verwenden. AWS CLI Wenn das Geheimnis in einem anderen Konto vorhanden ist, muss der vollständige ARN des Geheimnisses angegeben werden. Bei Verwendung von muss immer der vollständige ARN des Geheimnisses angegeben werden. AWS Management Console

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition, welche die erforderlichen Parameter zeigt:

Ersetzen Sie *`private-repo`* durch den Hostnamen des privaten Repositorys und *`private-image`* durch den Image-Namen.

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

> **Note**
>
> Eine weitere Methode zur Aktivierung der privaten Registrierungsauthentifikation verwendet Umgebungsvariablen von Amazon-ECS-Container-Agenten für die Authentifizierung bei privaten Registrierungen. Diese Methode wird nur für Aufgaben unterstützt, die auf Amazon-EC2-Instances gehostet werden. Weitere Informationen finden Sie unter Konfiguration von Amazon ECS-Container-Instances für private Docker-Images .

Um die private Registrierung zu verwenden

1. Die Aufgabendefinition muss über eine Aufgabenausführungsrolle verfügen. Auf diese Weise kann der Container-Agent das Container-Image abrufen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

   Um Zugriff auf die Secrets zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur Aufgabendefinitionsrolle hinzufügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

   - `secretsmanager:GetSecretValue`
   - `kms:Decrypt`: Nur erforderlich, wenn Ihr Schlüssel einen benutzerdefinierten KMS-Schlüssel verwendet und nicht den Standard-Schlüssel. Der Amazon-Ressourcenname (ARN) für Ihren benutzerdefinierten Schlüssel muss als Ressource hinzugefügt werden.

   Das folgende Beispiel einer Inline-Richtlinie fügt die Berechtigungen hinzu:

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "kms:Decrypt",
                   "secretsmanager:GetSecretValue"
               ],
               "Resource": [

    "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                   "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
               ]
           }
       ]
   }
   ```

2. Wird verwendet AWS Secrets Manager , um ein Geheimnis für Ihre privaten Registrierungsdaten zu erstellen. Informationen zum Erstellen eines Geheimnisses finden Sie unter Create an AWS Secrets Manager Secret im AWS Secrets Manager Benutzerhandbuch.

   Geben Sie Ihre Anmeldedaten für die private Registrierung im folgenden Format ein:

```
{
  "username" : "privateRegistryUsername",
  "password" : "privateRegistryPassword"
}
```

3. Eine Aufgabendefinition registrieren. Weitere Informationen finden Sie unter the section called “Erstellen einer Aufgabendefinition mit der Konsole”.

Übergeben Sie eine einzelne Umgebungsvariable an einen Amazon ECS-Container

> **Important**
>
> Wir empfehlen, Ihre sensiblen Daten entweder in AWS Secrets Manager Secrets- oder AWS Systems Manager Parameter Store-Parametern zu speichern. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.
> Die in der Aufgabendefinition angegebenen Umgebungsvariablen sind für alle Benutzer und Rollen lesbar, die die `DescribeTaskDefinition`-Aktion für die Aufgabendefinition durchführen dürfen.

Sie können Umgebungsvariablen auf folgende Weise an Ihre Container übergeben:

- Individuell mit dem `environment`-Containerdefinitionsparameter. Dies wird auf die Option `--env` abgebildet, um auf die docker run zu verweisen.
- Verwenden Sie den Container-Definitionsparameter `environmentFiles`, um eine oder mehrere Dateien aufzulisten, die die Umgebungsvariablen enthalten. Die Datei muss in Amazon S3 gehostet werden. Dies wird auf die Option `--env-file` abgebildet, um auf die docker run zu verweisen.

Im Folgenden finden Sie ein Ausschnitt aus einer Aufgabendefinition, in dem gezeigt wird, wie einzelne Umgebungsvariablen angegeben werden.

```
{
    "family": "",
    "containerDefinitions": [
```

```
        {
            "name": "",
            "image": "",
            ...
            "environment": [
                {
                    "name": "variable",
                    "value": "value"
                }
            ],
            ...
        }
    ],
    ...
}
```

Umgebungsvariablen an einen Amazon ECS-Container übergeben

> **Important**
>
> Wir empfehlen, Ihre sensiblen Daten entweder in AWS Secrets Manager Secrets- oder AWS Systems Manager Parameter Store-Parametern zu speichern. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.
> Umgebungsvariablen-Dateien sind Objekte in Amazon S3, und es gelten alle Sicherheitsüberlegungen von Amazon S3.
> Sie können den `environmentFiles` Parameter nicht für Windows-Container und Windows-Container für Fargate verwenden.

Sie können eine Umgebungsvariablendatei erstellen und in Amazon S3 speichern, um Umgebungsvariablen an Ihren Container zu übergeben.

Durch das Angeben von Umgebungsvariablen in einer Datei können Sie Umgebungsvariablen gesammelt einfügen. Geben Sie innerhalb der Containerdefinition das Objekt `environmentFiles` mit einer Liste von Amazon S3-Buckets an, die Ihre Umgebungsvariablendateien enthalten.

Amazon ECS erzwingt keine Größenbeschränkung für die Umgebungsvariablen, aber eine große Umgebungsvariablendatei kann den Festplattenspeicher belegen. Jede Aufgabe, die eine Umgebungsvariablendatei verwendet, bewirkt, dass eine Kopie der Datei auf den Datenträger heruntergeladen wird. Amazon ECS entfernt die Datei im Rahmen der Aufgabenbereinigung.

Informationen zu den unterstützten Umgebungsvariablen finden Sie unter Erweiterte Container-Definitionsparameter – Umgebung.

Berücksichtigen Sie Folgendes, wenn eine Umgebungsvariablendatei in einer Containerdefinition angegeben wird.

- Für Amazon-ECS-Aufgaben auf Amazon EC2 benötigen Ihre Container-Instances die Container-Agenten-Version `1.39.0` oder höher, um dieses Feature verwenden zu können. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.
- Für Amazon ECS-Aufgaben auf AWS Fargate müssen Ihre Aufgaben die Plattformversion `1.4.0` oder höher (Linux) verwenden, um diese Funktion nutzen zu können. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

  Stellen Sie sicher, dass die Variable für die Betriebssystem-Plattform unterstützt wird. Weitere Informationen finden Sie unter the section called “Containerdefinitionen” und the section called “Andere Parameter der Aufgabendefinition”.
- Die Datei muss die `.env`-Dateierweiterung und die UTF-8-Kodierung verwenden.
- Es gibt ein Limit von 10 Dateien pro Aufgabendefinition.
- Jede Zeile in einer Umgebungsdatei muss eine Umgebungsvariable im Format `VARIABLE=VALUE` enthalten. Leerzeichen oder Anführungszeichen werden als Teil der Werte für Amazon-ECS-Dateien einbezogen. Zeilen, die mit # beginnen, werden als Kommentare behandelt und ignoriert. Weitere Informationen zur Syntax der Umgebungsvariablen finden Sie unter Deklarieren von Standardumgebungsvariablen in Datei.

  Im Folgenden finden Sie die entsprechende Syntax.

  ```
  #This is a comment and will be ignored
  VARIABLE=VALUE
  ENVIRONMENT=PRODUCTION
  ```
- Wenn Umgebungsvariablen mit dem `environment`-Parameter in einer Containerdefinition angegeben sind, haben sie Vorrang vor den Variablen, die in einer Umgebungsdatei enthalten sind.
- Wenn mehrere Umgebungsdateien angegeben sind und sie dieselbe Variable enthalten, werden sie in der Reihenfolge ihres Eintrags verarbeitet. Dies bedeutet, dass der erste Wert der Variablen verwendet wird und nachfolgende Werte doppelter Variablen ignoriert werden. Es wird empfohlen, eindeutige Variablennamen zu verwenden.

- Wenn eine Umgebungsdatei als Container-Überschreibung angegeben wird, wird sie verwendet. Darüber hinaus werden alle anderen Umgebungsdateien ignoriert, die in der Containerdefinition angegeben sind.
- Die folgenden Regeln gelten für den Starttyp Fargate:
  - Die Datei wird wie eine native Docker-Umgebungsdatei behandelt.
  - Es gibt keine Unterstützung für Shell-Escape-Handling.
  - Der Container-Einstiegspunkt interpretiert die `VARIABLE`-Werte.

Erforderliche IAM-Berechtigungen

Die Amazon-ECS-Aufgabenausführungsrolle ist erforderlich, um diese Funktion zu verwenden. Auf diese Weise kann der Container-Agent die Umgebungsvariablendatei von Amazon S3 abrufen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Um Zugriff auf die Amazon S3-Objekte zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen manuell als eingebundene Richtlinie zur Aufgabendefinitionsrolle hinzufügen. Verwenden Sie den `Resource`-Parameter, um die Berechtigung auf die Amazon S3-Buckets zu erweitern, die die Dateien der Umgebungsvariablen enthalten. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- `s3:GetObject`
- `s3:GetBucketLocation`

Im folgenden Beispiel werden diese Berechtigungen einer Inline-Richtlinie hinzugefügt.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/folder_name/env_file_name"
      ]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket"
      ]
    }
  ]
}
```

Beispiel

Im Folgenden finden Sie ein Ausschnitt aus einer Aufgabendefinition, in dem gezeigt wird, wie eine Umgebungsvariablendatei angegeben wird.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environmentFiles": [
                {
                    "value": "arn:aws:s3:::s3_bucket_name/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Übergeben Sie sensible Daten an einen Amazon ECS-Container

Sie können vertrauliche Daten, wie z. B. Anmeldeinformationen für eine Datenbank, sicher in Ihren Container übergeben.

Sie können Secrets Manager oder als Parameter im Systems Manager Parameter Store verwenden, um das Geheimnis zu speichern.

Sie können Geheimnisse programmgesteuert aus der Anwendung oder mithilfe von Umgebungsvariablen abrufen.

Speichern Sie zunächst die sensiblen Daten als Geheimnis in Secrets Manager oder als Parameter im Systems Manager Parameter Store. Verwenden Sie dann eine der folgenden Möglichkeiten, um das Geheimnis für den Container verfügbar zu machen.

Themen

Bewährte Methoden für die Verwaltung von Geheimnissen in Amazon ECS

Secrets, wie API-Schlüssel und Datenbankanmeldeinformationen, werden häufig von Anwendungen verwendet, um auf andere Systeme zuzugreifen. Sie bestehen häufig aus einem Benutzernamen und einem Passwort, einem Zertifikat oder einem API-Schlüssel. Der Zugriff auf diese Secrets sollte auf bestimmte IAM-Prinzipale beschränkt werden, die IAM verwenden und zur Laufzeit in Container eingespeist werden.

Secrets können nahtlos aus AWS Secrets Manager einem Amazon EC2 Systems Manager Parameter Store in Container eingefügt werden. Auf diese Secrets kann in Ihrer Aufgabe wie folgt verwiesen werden.

1. Sie werden als Umgebungsvariablen referenziert, die den `secrets`-Container-Definitionsparameter verwenden.
2. Sie werden als `secretOptions` bezeichnet, wenn Ihre Protokollierungsplattform eine Authentifizierung erfordert. Weitere Informationen finden Sie unter Konfigurationsoptionen für die Protokollierung.
3. Sie werden als Secrets bezeichnet, die von Images abgerufen werden, die den `repositoryCredentials`-Container-Definitionsparameter verwenden, wenn die Registrierung,

aus der der Container abgerufen wird, eine Authentifizierung erfordert. Verwenden Sie diese Methode, wenn Sie Images aus Amazon ECR Public Gallery abrufen. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Aufgaben.

Empfehlungen zu Geheimnissen

Wir empfehlen Ihnen, bei der Einrichtung der Verwaltung von Secrets wie folgt vorzugehen.

Verwenden Sie AWS Secrets Manager unseren Amazon EC2 Systems Manager Parameter Store zum Speichern geheimer Materialien

Sie sollten API-Schlüssel, Datenbankanmeldedaten und andere geheime Materialien sicher in AWS Secrets Manager oder als verschlüsselte Parameter im Amazon EC2 Systems Manager Parameter Store speichern. Diese Dienste ähneln sich, da es sich bei beiden um verwaltete Schlüsselwertspeicher handelt, die AWS KMS zur Verschlüsselung sensibler Daten verwendet werden. AWS Secrets Manager beinhaltet jedoch auch die Möglichkeit, Geheimnisse automatisch zu rotieren, zufällige Geheimnisse zu generieren und Geheimnisse zwischen Konten auszutauschen. AWS Wenn Sie diese Features als wichtig einstufen, verwenden Sie AWS Secrets Manager , ansonsten verwenden Sie verschlüsselte Parameter.

> **Note**
>
> Aufgaben, die auf ein Geheimnis aus AWS Secrets Manager oder dem Amazon EC2 Systems Manager Parameter Store verweisen, erfordern eine Aufgabenausführungsrolle mit einer Richtlinie, die Amazon ECS Zugriff auf das gewünschte Geheimnis und, falls zutreffend, auf den AWS KMS Schlüssel gewährt, der zum Verschlüsseln und Entschlüsseln dieses Geheimnisses verwendet wird.

> **Important**
>
> Secrets, auf die in Aufgaben verwiesen wird, werden nicht automatisch rotiert. Wenn sich Ihr Secret ändert, müssen Sie eine neue Bereitstellung erzwingen oder eine neue Aufgabe starten, um den neuesten Secret-Wert abzurufen. Weitere Informationen finden Sie unter den folgenden Themen:
>
> - AWS Secrets Manager: Daten als Umgebungsvariablen einfügen
> - Amazon EC2 Systems Manager Parameter Store: Daten als Umgebungsvariablen einfügen

Daten aus einem verschlüsselten Amazon S3 S3-Bucket abrufen

Da der Wert von Umgebungsvariablen versehentlich in Protokollen nach außen dringen kann und bei der Ausführung von `docker inspect` aufgedeckt wird, sollten Sie Secrets in einem verschlüsselten Amazon-S3-Bucket speichern und Aufgabenrollen verwenden, um den Zugriff auf diese Secrets zu beschränken. Wenn Sie dies tun, muss Ihre Anwendung so geschrieben werden, dass sie das Secret aus dem Amazon-S3-Bucket liest. Anweisungen dazu finden Sie unter Festlegen des standardmäßigen serverseitigen Verschlüsselungsverhaltens für Amazon-S3-Buckets.

Das Secret mit Hilfe eines Beiwagen-Containers in ein Volume mounten

Da bei Umgebungsvariablen ein erhöhtes Risiko von Datenlecks besteht, sollten Sie einen Sidecar-Container verwenden, der Ihre Secrets ausliest AWS Secrets Manager und auf ein gemeinsam genutztes Volume schreibt. Dieser Container kann vor dem Anwendungscontainer ausgeführt und beendet werden, indem Sie Amazon-ECS-Container-Anordnungen verwenden. Wenn Sie dies tun, mountet der Anwendungscontainer anschließend das Volume, auf dem das Secret geschrieben wurde. Wie bei der Amazon-S3-Bucket-Methode muss Ihre Anwendung so geschrieben werden, dass sie das Secret aus dem gemeinsam genutzten Volume liest. Da das Volume auf die Aufgabe beschränkt ist, wird das Volume nach dem Beenden der Aufgabe automatisch gelöscht. Ein Beispiel für einen Beiwagen-Container finden Sie im Projekt aws-secret-sidecar-injector.

> **Note**
>
> In Amazon EC2 kann das Volume, auf welches das Secret geschrieben wird, mit einem vom Kunden verwalteten AWS KMS -Schlüssel verschlüsselt werden. Bei aktivierter AWS Fargate Option wird der Datenträgerspeicher automatisch mithilfe eines vom Service verwalteten Schlüssels verschlüsselt.

Weitere Ressourcen

- Weitergabe von Secrets an Container in einer Amazon-ECS-Aufgabe
- Chamber ist ein Wrapper zum Speichern von Secrets im Amazon EC2 Systems Manager Parameter Store

Secrets Manager Manager-Geheimnisse programmgesteuert in Amazon ECS abrufen

Verwenden Sie Secrets Manager, um vertrauliche Daten zu schützen und Datenbankanmeldeinformationen, API-Schlüssel und andere Geheimnisse während ihres gesamten Lebenszyklus zu rotieren, zu verwalten und abzurufen.

Anstatt vertrauliche Informationen in Ihrer Anwendung im Klartext fest zu codieren, können Sie Secrets Manager verwenden, um die vertraulichen Daten zu speichern.

Wir empfehlen diese Methode zum Abrufen vertraulicher Daten, da die Anwendung automatisch die neueste Version des Secrets-Manager-Geheimnisses abruft, wenn das Secrets-Manager-Geheimnis anschließend aktualisiert wird.

Erstellen Sie ein Geheimnis in Secrets Manager. Nachdem Sie ein Secrets-Manager-Geheimnis erstellt haben, aktualisieren Sie Ihren Anwendungscode, um das Geheimnis abzurufen.

Lesen Sie die folgenden Überlegungen, bevor Sie vertrauliche Daten in Secrets Manager sichern.

- Es werden nur Geheimnisse unterstützt, die Textdaten speichern, bei denen es sich um Geheimnisse handelt, die mit dem `SecretString` CreateSecretAPI-Parameter erstellt wurden. Geheimnisse, die Binärdaten speichern, bei denen es sich um Geheimnisse handelt, die mit dem `SecretBinary` CreateSecretAPI-Parameter erstellt wurden, werden nicht unterstützt.
- Verwenden Sie Schnittstellen-VPC-Endpunkte, um die Sicherheitskontrollen zu verbessern. Sie müssen die Schnittstellen-VPC-Endpunkte für den Secrets Manager erstellen. Informationen über den VPC-Endpunkt finden Sie unter VPC-Endpunkte erstellen im AWS Secrets Manager - Benutzerhandbuch.
- Die von Ihrer Aufgabe verwendete VPC muss die DNS-Auflösung verwenden.

Erforderliche IAM-Berechtigungen

Zum Verwenden dieses Features benötigen Sie die Amazon-ECS-Aufgabenrolle und müssen in Ihrer Aufgabendefinition auf sie verweisen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

Um Zugriff auf die Secrets-Manager-Geheimnisse zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen manuell zur Aufgabendefinitionsrolle hinzufügen. Informationen zum Verwalten von Berechtigungen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

- `secretsmanager:GetSecretValue` – Erforderlich, wenn Sie auf ein Secrets Manager-Geheimnis verweisen. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
      ]
    }
  ]
}
```

Erstellen des Secrets-Manager-Geheimnisses

Sie können die Secrets Manager-Konsole verwenden, um ein Secret für Ihre sensiblen Daten zu erstellen. Informationen zum Erstellen von Geheimnissen finden Sie unter Erstellen eines AWS Secrets Manager -Geheimnisses im AWS Secrets Manager -Benutzerhandbuch.

Aktualisieren Sie Ihre Anwendung, um Secrets-Manager-Geheimnisse programmgesteuert abzurufen

Sie können Geheimnisse abrufen, indem Sie die Secrets-Manager-APIs direkt aus Ihrer Anwendung aufrufen. Weitere Informationen finden Sie AWS Secrets Manager im AWS Secrets Manager Benutzerhandbuch unter Geheimnisse abrufen von.

Informationen zum Abrufen der in der AWS Secrets Manager gespeicherten vertraulichen Daten finden Sie unter Codebeispiele für die AWS Secrets Manager Verwendung von AWS SDKs in der AWS SDK-Codebeispiel-Codebibliothek.

Rufen Sie Systems Manager Manager-Parameter ab und speichern Sie Geheimnisse programmgesteuert in Amazon ECS

Systems Manager Parameter Store ermöglicht die sichere Speicherung und Verwaltung von Geheimnissen. Sie können Daten wie Passwörter, Datenbankzeichenfolgen, EC2-Instanz-IDs und AMI-IDs sowie Lizenzcodes als Parameterwerte speichern. Sie können Werte als Klartext oder als verschlüsselte Daten speichern.

Anstatt vertrauliche Informationen in Ihrer Anwendung im Klartext fest zu codieren, können Sie Secrets Manager verwenden, um die vertraulichen Daten zu speichern.

Wir empfehlen diese Methode zum Abrufen vertraulicher Daten, da die Anwendung bei einer späteren Aktualisierung des Systems Manager Manager-Parameterspeicher-Parameters automatisch die neueste Version abruft.

Erstellen Sie ein Geheimnis in Secrets Manager. Nachdem Sie ein Secrets-Manager-Geheimnis erstellt haben, aktualisieren Sie Ihren Anwendungscode, um das Geheimnis abzurufen.

Lesen Sie die folgenden Überlegungen, bevor Sie sensible Daten im Systems Manager Parameter Store sichern.

- Es werden nur Geheimnisse unterstützt, die Textdaten speichern. Geheimnisse zum Speichern von Binärdaten werden nicht unterstützt.
- Verwenden Sie Schnittstellen-VPC-Endpunkte, um die Sicherheitskontrollen zu verbessern.
- Die von Ihrer Aufgabe verwendete VPC muss die DNS-Auflösung verwenden.

Erforderliche IAM-Berechtigungen

Zum Verwenden dieses Features benötigen Sie die Amazon-ECS-Aufgabenrolle und müssen in Ihrer Aufgabendefinition auf sie verweisen. Dies ermöglicht dem Container-Agent das Abrufen der erforderlichen Systems-Manager-Ressourcen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

> **Important**
>
> Für Aufgaben, die den Starttyp EC2 verwenden, müssen Sie die ECS-Agent-Konfigurationsvariable `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` verwenden, um diese Funktion verwenden zu können. Sie können sie während der Erstellung

der Container-Instance zur Datei `./etc/ecs/ecs.config` hinzufügen oder sie zu einer vorhandenen Instance hinzufügen und dann den ECS-Agenten neu starten. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Um Zugriff auf die von Ihnen erstellten Systems-Manager-Parameter-Store-Parameter zu erhalten, fügen Sie der Aufgabenausführungsrolle manuell die folgenden Berechtigungen als Inline Richtlinie hinzu. Informationen zum Verwalten von Berechtigungen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

- `ssm:GetParameters` – Erforderlich, wenn in einer Aufgabendefinition auf einen Parameter-Store-Parameter von Systems Manager verwiesen wird. Fügt die Berechtigung zum Abrufen von Systems-Manager-Parametern hinzu.
- `secretsmanager:GetSecretValue` – Erforderlich, wenn Sie direkt auf ein Secrets-Manager-Geheimnis verweisen oder wenn der Parameter Systems Manager Parameter Store in einer Aufgabendefinition auf ein Secrets-Manager-Geheimnis verweist. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.
- `kms:Decrypt` – Nur erforderlich, wenn Ihr Geheimnis einen kundenverwalteten Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren benutzerdefinierten Schlüssel sollte als Ressource hinzugefügt werden. Fügt die Berechtigung zum Entschlüsseln des vom Kunden verwalteten Schlüssels hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
```

```
    }
  ]
}
```

Erstellen des -Parameters

Sie können die Systems-Manager-Konsole verwenden, um einen Parameter des Systems Manager Parameter Stores für Ihre sensiblen Daten zu erstellen. Weitere Informationen finden Sie im AWS Systems Manager -Benutzerhandbuch unter Erstellen eines Systems-Manager-Parameters (Konsole) oder Erstellen eines Systems-Manager-Parameters (AWS CLI).

Aktualisieren Sie Ihre Anwendung, um die Geheimnisse des Systems-Manager-Parameter-Speichers programmgesteuert abzurufen

Informationen zum Abrufen der sensiblen Daten, die im Parameter Store-Parameter von Systems Manager gespeichert sind, finden Sie unter Codebeispiele für Systems Manager mit AWS SDKs in der AWS SDK-Codebeispiel-Codebibliothek.

Secrets Manager Manager-Geheimnisse über Amazon ECS-Umgebungsvariablen abrufen

Wenn Sie ein Secret als Umgebungsvariable einfügen, können Sie den vollständigen Inhalt eines Secrets, eines bestimmten JSON-Schlüssels innerhalb eines Secrets oder einer bestimmten Version eines einzufügenden Secrets angeben. Das hilft Ihnen, die sensiblen Daten zu steuern, die Ihrem Container zur Verfügung gestellt werden. Weitere Informationen zur geheimen Versionsverwaltung finden Sie unter Schlüsselbegriffe und -konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

Folgendes sollte beachtet werden, wenn eine Umgebungsvariable verwendet wird, um ein Secrets Manager Manager-Geheimnis in einen Container einzufügen.

- Sensible Daten werden beim ersten Start des Containers an diesen übergeben. Wenn das Secret anschließend aktualisiert oder rotiert wird, erhält der Container nicht automatisch den aktualisierten Wert. Sie müssen eine neue Aufgabe starten. Alternativ können Sie, wenn Ihre Aufgabe Teil eines Services ist, den Service aktualisieren und die Option Force new deployment (Neue Bereitstellung erzwingen) auswählen, um den Service zu zwingen, eine neue Aufgabe zu starten.
- Bei Amazon ECS-Aufgaben sollte Folgendes beachtet werden: AWS Fargate
  - Um den vollständigen Inhalt eines Secrets als Umgebungsvariable oder in eine Protokollkonfiguration einzufügen, müssen Sie die Plattformversion `1.3.0` oder höher

verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

- Um einen bestimmten JSON-Schlüssel oder eine Version eines Geheimnisses als Umgebungsvariable oder in eine Protokollkonfiguration einzufügen, müssen Sie die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` (Windows) verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

- Bei Amazon ECS-Aufgaben auf EC2 sollte Folgendes berücksichtigt werden:
  - Um einen geheimen Schlüssel mithilfe eines bestimmten JSON-Schlüssels oder einer Secret-Version einzufügen, muss Ihre Container-Instance Version `1.37.0` oder höher des Container-Agenten haben. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

    Um den vollständigen Inhalt eines Secrets als Umgebungsvariable einzufügen oder ein Secret in eine Protokollkonfiguration einzufügen, muss Ihre Container-Instance Version `1.22.0` oder höher des Container-Agenten haben.

- Verwenden Sie VPC-Endpunkte mit Schnittstellen, um die Sicherheitskontrollen zu verbessern, und stellen Sie über ein privates Subnetz eine Verbindung zu Secrets Manager her. Sie müssen die Schnittstellen-VPC-Endpunkte für den Secrets Manager erstellen. Informationen über den VPC-Endpunkt finden Sie unter VPC-Endpunkte erstellen im AWS Secrets Manager - Benutzerhandbuch. Weitere Informationen zur Verwendung von Secrets Manager und Amazon VPC finden Sie unter So stellen Sie eine Verbindung zum Secrets Manager Manager-Service in einer Amazon VPC her.

- Für Windows-Aufgaben, die für die Verwendung des `awslogs`-Protokolltreibers konfiguriert sind, müssen Sie auch die `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE`-Umgebungsvariable für die Container-Instance festlegen. Dies kann mit Benutzerdaten unter Verwendung der folgenden Syntax erfolgen:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

IAM-Berechtigungen

Zum Verwenden dieses Features benötigen Sie die Amazon-ECS-Aufgabenausführungsrolle und müssen in Ihrer Aufgabendefinition auf sie verweisen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Um Zugriff auf die Secrets Manager-Secrets zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen manuell als eingebundene Richtlinie zur Aufgabendefinitionsrolle hinzufügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- `secretsmanager:GetSecretValue`: Erforderlich, wenn Sie auf ein Secrets Manager-Secret verweisen. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.
- `kms:Decrypt` – Nur erforderlich, wenn Ihr Geheimnis einen kundenverwalteten Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren vom Benutzer verwalteten Schlüssel sollte als Ressource hinzugefügt werden. Fügt die Berechtigung zum Entschlüsseln des vom Kunden verwalteten Schlüssels hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Erstellen des AWS Secrets Manager -Geheimnisses

Sie können die Secrets Manager-Konsole verwenden, um ein Secret für Ihre sensiblen Daten zu erstellen. Weitere Informationen finden Sie unter Create an AWS Secrets Manager Secret im AWS Secrets Manager Benutzerhandbuch.

Fügen Sie die Umgebungsvariable zur Container-Definition hinzu

Innerhalb der Containerdefinition können Sie Folgendes angeben:

- Das `secrets`-Objekt, das den Namen der Umgebungsvariablen enthält, die im Container festgelegt werden soll
- Der Amazon-Ressourcenname (ARN) des Secrets Manager-Secrets
- Zusätzliche Parameter, die die sensiblen Daten enthalten, die dem Container angezeigt werden sollen

Das folgende Beispiel zeigt die vollständige Syntax, die für das Secrets Manager-Secret angegeben werden muss.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-stage:version-id
```

Im folgenden Abschnitt werden die zusätzlichen Parameter beschrieben. Diese Parameter sind optional, aber wenn Sie sie nicht verwenden, müssen Sie die Doppelpunkte einschließen, damit `:` die Standardwerte verwendet. Beispiele finden Sie unten für weiteren Kontext.

`json-key`

Gibt den Namen des Schlüssels in einem Schlüssel-Wert-Paar mit dem Wert an, den Sie als Umgebungsvariablenwert festlegen möchten. Nur Werte im JSON-Format werden unterstützt. Wenn Sie keinen JSON-Schlüssel angeben, wird der vollständige Inhalt des Secrets verwendet.

`version-stage`

Gibt die Phasenbeschriftung der Version eines Secrets an, die Sie verwenden möchten. Wenn eine Versionsphasenbeschriftung angegeben ist, können Sie keine Versions-ID angeben. Wenn keine Versionsphase angegeben wird, besteht das Standardverhalten darin, das Secret Schlüssel mit der `AWSCURRENT`-Phasenbeschriftung abzurufen.

Phasenbeschriftungen werden verwendet, um verschiedene Versionen eines Secrets zu verfolgen, wenn sie aktualisiert oder rotiert werden. Jede Version eines Secrets hat eine oder mehrere Phasenbeschriftungen und eine ID. Weitere Informationen erhalten Sie unter Zentrale Begriffe und Konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

`version-id`

Gibt die eindeutige ID der Version des Secrets an, die Sie verwenden möchten. Wenn eine Versions-ID angegeben wird, können Sie keine Versionsphasenbeschriftung angeben. Wenn keine Versions-ID angegeben wird, besteht das Standardverhalten darin, den geheimen Schlüssel mit der `AWSCURRENT`-Phasenbeschriftung abzurufen.

Versions-IDs werden verwendet, um verschiedene Versionen eines Secrets zu verfolgen, wenn sie aktualisiert oder rotiert werden. Jede Version eines Secrets hat eine ID. Weitere Informationen erhalten Sie unter Zentrale Begriffe und Konzepte für AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

Beispiel-Containerdefinitionen

Die folgenden Beispiele zeigen, wie Sie auf Secrets Manager-Secrets in Ihren Containerdefinitionen verweisen können.

Example Verweisen auf ein vollständiges Secret

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition mit dem Format beim Verweisen auf den vollständigen Text eines Secrets Manager-Secret.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Um innerhalb des Containers auf den Wert dieses Geheimnisses zuzugreifen, müssten Sie den `$environment_variable_name` aufrufen.

Example Verweisen auf einen bestimmten Schlüssel innerhalb eines Secrets

Im Folgenden wird eine Beispielausgabe eines get-secret-value-Befehls gezeigt, der den Inhalt eines Secrets zusammen mit der Versionsphasenbeschriftung und der zugehörigen Versions-ID anzeigt.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981ddEXAMPLE",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Verweisen Sie auf einen bestimmten Schlüssel aus der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example Verweisen auf eine bestimmte Secret-Version

Im Folgenden wird eine Beispielausgabe eines describe-secret -Befehls gezeigt, der den unverschlüsselten Inhalt eines Secrets zusammen mit den Metadaten für alle Versionen des Secrets anzeigt.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
```

```
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981ddEXAMPLE": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE": [
            "AWSPREVIOUS"
        ]
    }
}
```

Verweisen Sie auf eine bestimmte Versionsphasenbeschriftung aus der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Verweisen Sie auf eine bestimmte Versions-ID der vorherigen Ausgabe in einer Containerdefinition, indem Sie den Schlüsselnamen am Ende des ARN angeben.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Example Verweisen auf einen bestimmten Schlüssel und eine Versionsphasenbeschriftung eines Secrets

Im Folgenden wird gezeigt, wie Sie sowohl auf einen bestimmten Schlüssel innerhalb eines Secrets als auch auf eine bestimmte Versionsphasenbeschriftung verweisen.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Verwenden Sie die folgende Syntax, um einen bestimmten Schlüssel und eine Versions-ID anzugeben.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Informationen zum Erstellen einer Aufgabendefinition mit dem in einer Umgebungsvariablen angegebenen Geheimzahl finden Sie unterErstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Rufen Sie Systems Manager Manager-Parameter über Amazon ECS-Umgebungsvariablen ab

Amazon ECS ermöglicht es Ihnen, vertrauliche Daten in Ihre Container einzufügen, indem Sie Ihre sensiblen Daten in AWS Systems Manager Parameter Store-Parametern speichern und sie dann in Ihrer Container-Definition referenzieren.

Beachten Sie Folgendes, wenn Sie eine Umgebungsvariable verwenden, um ein Systems Manager Manager-Geheimnis in einen Container einzufügen.

- Sensible Daten werden beim ersten Start des Containers an diesen übergeben. Wenn das Secret anschließend aktualisiert oder rotiert wird, erhält der Container nicht automatisch den aktualisierten Wert. Sie müssen eine neue Aufgabe starten. Alternativ können Sie, wenn Ihre Aufgabe Teil eines Services ist, den Service aktualisieren und die Option Force new deployment (Neue Bereitstellung erzwingen) auswählen, um den Service zu zwingen, eine neue Aufgabe zu starten.
- Bei Amazon ECS-Aufgaben sollte Folgendes beachtet werden: AWS Fargate
  - Um den vollständigen Inhalt eines Secrets als Umgebungsvariable oder in eine Protokollkonfiguration einzufügen, müssen Sie die Plattformversion `1.3.0` oder höher verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
  - Um einen bestimmten JSON-Schlüssel oder eine Version eines Geheimnisses als Umgebungsvariable oder in eine Protokollkonfiguration einzufügen, müssen Sie die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` (Windows) verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
- Bei Amazon ECS-Aufgaben auf EC2 sollte Folgendes berücksichtigt werden:
  - Um einen geheimen Schlüssel mithilfe eines bestimmten JSON-Schlüssels oder einer Secret-Version einzufügen, muss Ihre Container-Instance Version `1.37.0` oder höher des Container-Agenten haben. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

    Um den vollständigen Inhalt eines Secrets als Umgebungsvariable einzufügen oder ein Secret in eine Protokollkonfiguration einzufügen, muss Ihre Container-Instance Version `1.22.0` oder höher des Container-Agenten haben.
- Verwenden Sie Schnittstellen-VPC-Endpunkte, um die Sicherheitskontrollen zu verbessern. Sie müssen die VPC-Schnittstellen-Endpunkte für Systems Manager erstellen. Informationen über den VPC-Endpunkt finden Sie unter VPC-Endpunkte erstellen im AWS Systems Manager - Benutzerhandbuch.
- Für Windows-Aufgaben, die für die Verwendung des `awslogs`-Protokolltreibers konfiguriert sind, müssen Sie auch die `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE`-Umgebungsvariable für die Container-Instance festlegen. Dies kann mit Benutzerdaten unter Verwendung der folgenden Syntax erfolgen:

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

IAM-Berechtigungen

Zum Verwenden dieses Features benötigen Sie die Amazon-ECS-Aufgabenausführungsrolle und müssen in Ihrer Aufgabendefinition auf sie verweisen. Dies ermöglicht dem Container-Agent das Abrufen der erforderlichen Systems-Manager-Ressourcen. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Important

Für Aufgaben, die den Starttyp EC2 verwenden, müssen Sie die ECS-Agent-Konfigurationsvariable `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` verwenden, um diese Funktion verwenden zu können. Sie können sie während der Erstellung der Container-Instance zur Datei `./etc/ecs/ecs.config` hinzufügen oder sie zu einer vorhandenen Instance hinzufügen und dann den ECS-Agenten neu starten. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Um Zugriff auf die von Ihnen erstellten Parameter Store-Parameter von Systems Manager zu gewähren, fügen Sie der Aufgabenausführungsrolle manuell die folgenden Berechtigungen hinzu. Informationen zum Verwalten von Berechtigungen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

- `ssm:GetParameters` – Erforderlich, wenn in einer Aufgabendefinition auf einen Parameter-Store-Parameter von Systems Manager verwiesen wird. Fügt die Berechtigung zum Abrufen von Systems-Manager-Parametern hinzu.
- `secretsmanager:GetSecretValue` – Erforderlich, wenn Sie direkt auf ein Secrets-Manager-Geheimnis verweisen oder wenn der Parameter Systems Manager Parameter Store in einer Aufgabendefinition auf ein Secrets-Manager-Geheimnis verweist. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.

- `kms:Decrypt` – Nur erforderlich, wenn Ihr Geheimnis einen kundenverwalteten Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren benutzerdefinierten Schlüssel sollte als Ressource hinzugefügt werden. Fügt die Berechtigung zum Entschlüsseln des vom Kunden verwalteten Schlüssels hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Erstellen Sie den Systems Manager Manager-Parameter

Sie können die Systems-Manager-Konsole verwenden, um einen Parameter des Systems Manager Parameter Stores für Ihre sensiblen Daten zu erstellen. Weitere Informationen finden Sie im AWS Systems Manager -Benutzerhandbuch unter Erstellen eines Systems-Manager-Parameters (Konsole) oder Erstellen eines Systems-Manager-Parameters (AWS CLI).

Fügen Sie die Umgebungsvariable zur Container-Definition hinzu

Geben Sie in Ihrer Containerdefinition `secrets` mit dem Namen der im Container zu setzenden Umgebungsvariablen und dem Namen oder dem ARN des Systems Manager-Parameter Store-Parameters an, der die sensiblen Daten enthält, die dem Container präsentiert werden sollen. Weitere Informationen finden Sie unter `secrets`.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition mit dem Format beim Verweisen auf einen Systems Manager Parameter Store-Parameter. Wenn sich der Systems Manager Parameter Store-Parameter in der gleichen Region wie die Aufgabe befindet, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, geben Sie den vollen ARN an.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Hinweise zum Erstellen einer Aufgabendefinition mit dem in einer Umgebungsvariablen angegebenen geheimen Schlüssel finden Sie unterErstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Geheimnisse für die Amazon ECS-Protokollierungskonfiguration abrufen

Sie können den `secretOptions` Parameter in verwenden, um sensible Daten `logConfiguration` zu übergeben, die für die Protokollierung verwendet werden.

Sie können das Geheimnis in Secrets Manager oder Systems Manager speichern.

Verwenden Sie Secrets Manager

Bei der Angabe von `logConfiguration` können Sie `secretOptions` in Ihrer Containerdefinition mit dem Namen der im Container festzulegenden Protokolltreiberoption und dem vollständigen ARN des Secrets Manager-Secrets angeben, in dem die sensiblen Daten enthalten sind, die dem Container zur Verfügung gestellt werden sollen.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition mit dem Format beim Verweisen auf ein Secrets Manager-Secret.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "splunk",
```

```
      "options": {
        "splunk-url": "https://your_splunk_instance:8088"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Verwenden von Systems Manager

Sie können vertrauliche Daten in eine Protokollkonfiguration einfügen. Beim Angeben von `logConfiguration` können Sie `secretOptions` in Ihrer Containerdefinition mit dem Namen der im Container festzulegenden Protokolltreiberoption und dem vollständigen ARN des Systems Manager-Parameter Store-Parameters angeben, in denen die sensiblen Daten enthalten sind, die dem Container präsentiert werden sollen.

> **Important**
>
> Wenn sich der Systems Manager Parameter Store-Parameter in der gleichen Region wie die Aufgabe befindet, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Parameters verwenden. Wenn der Parameter in einer anderen Region existiert, geben Sie den vollen ARN an.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition mit dem Format beim Verweisen auf einen Systems Manager Parameter Store-Parameter.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter:/parameter_name"
```

```
      }]
    }]
  }]
}
```

Angabe sensibler Daten mithilfe von Secrets Manager Manager-Geheimnissen in Amazon ECS

Amazon ECS ermöglicht es Ihnen, vertrauliche Daten in Ihre Container einzufügen, indem Sie Ihre sensiblen Daten in AWS Secrets Manager Geheimnissen speichern und dann in Ihrer Container-Definition auf sie verweisen. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

Erfahren Sie, wie Sie ein Secrets Manager-Geheimnis erstellen, in einer Amazon ECS-Aufgabendefinition auf das Geheimnis verweisen und dann überprüfen, ob es funktioniert hat, indem Sie die Umgebungsvariable in einem Container abfragen, der den Inhalt des Secrets anzeigt.

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen IAM-Berechtigungen, um die beschriebenen Secrets Manager- und Amazon ECS-Ressourcen zu erstellen.

Schritt 1: Erstellen eines Secrets-Manager-Secrets

Sie können die Secrets Manager-Konsole verwenden, um ein Secret für Ihre sensiblen Daten zu erstellen. In diesem Tutorial erstellen wir ein grundlegendes Secret zum Speichern eines Benutzernamens und eines Passworts zum späteren Verweisen in einem Container. Weitere Informationen finden Sie unter Erstellen eines Basic Secrets im AWS Secrets Manager - Benutzerhandbuch.

Die key/value pairs to be stored in this secret (Schlüssel/Wertpaare, die in diesem Geheimnis gespeichert werden sollen), stellen den Wert der Umgebungsvariablen in Ihrem Container am Ende des Tutorials dar.

Speichern Sie den Secret ARN (Geheimen ARN), um ihn in späteren Schritten in Ihrer IAM-Richtlinie für die Aufgabenausführung und der Aufgabendefinition zu verwenden.

Schritt 2: Aktualisieren Ihrer IAM-Rolle zur Ausführung von Aufgaben

Damit Amazon ECS sensible Daten aus Ihrem Secrets Manager-Secret abrufen kann, benötigen Sie die Amazon-ECS-Aufgabenausführungsrolle und müssen in Ihrer Aufgabendefinition darauf verweisen. Dies ermöglicht dem Container-Agent das Abrufen der erforderlichen Secrets Manager-Ressourcen. Weitere Informationen zum Erstellen Ihrer IAM-Aufgabenausführungsrolle finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

In den folgenden Schritten wird angenommen, dass Sie die IAM-Aufgabenausführungsrolle bereits erstellt und korrekt konfiguriert haben.

Aktualisieren Ihrer IAM-Aufgabenausführungsrolle

Verwenden Sie die IAM-Konsole zum Aktualisieren Ihrer Aufgabenausführungsrolle mit den erforderlichen Berechtigungen.

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen aus.
3. Durchsuchen Sie die Liste der Rollen für `ecsTaskExecutionRole` und wählen Sie diese aus.
4. Wählen Sie Permissions (Berechtigungen) und Add inline policy (Eingebundene Richtlinie hinzufügen) aus.
5. Wählen Sie die Registerkarte JSON aus und geben Sie den folgenden JSON-Text an. Stellen Sie sicher, dass Sie den vollständigen ARN des Secrets Manager-Secrets angeben, das Sie in Schritt 1 erstellt haben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [

 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value"
            ]
        }
    ]
}
```

6. Wählen Sie Richtlinie prüfen. Geben Sie unter Name `ECSSecretsTutorial` an und wählen Sie dann Create policy (Richtlinie erstellen).

Schritt 3: Erstellen einer Amazon-ECS-Aufgabendefinition

Sie können die Amazon-ECS-Konsole verwenden, um eine Aufgabendefinition zu erstellen, die auf ein Secrets Manager-Secret verweist.

So erstellen Sie eine Aufgabendefinition, die ein Secret angibt

Verwenden Sie die IAM-Konsole zum Aktualisieren Ihrer Aufgabenausführungsrolle mit den erforderlichen Berechtigungen.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
4. Geben Sie in das Feld JSON-Editor den folgenden JSON-Text für die Aufgabendefinition ein. Achten Sie darauf, dass Sie den vollständigen ARN des Secrets-Manager-Secrets, das Sie in Schritt 1 erstellt haben, und die IAM-Rolle für die Aufgabenausführung, die Sie im Schritt 2 aktualisiert haben, angeben. Wählen Sie Speichern.
5.

```
{
    "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
```

```
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ],
            "cpu": 10,
            "secrets": [
                {
                    "valueFrom":
 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value",
                    "name": "username_value"
                }
            ],
            "memory": 300,
            "image": "httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Wählen Sie Erstellen.

Schritt 4: Erstellen eines Amazon-ECS-Clusters

Verwenden Sie die Amazon-ECS-Konsole zum Erstellen eines Clusters, der eine Container-Instance zum Ausführen der Aufgabe enthält. Wenn Sie bereits über einen Cluster mit mindestens einer registrierten Container-Instance und den Ressourcen zum Ausführen einer Instance der für dieses Tutorial erstellten Aufgabendefinition verfügen, können Sie direkt zum nächsten Schritt gehen.

Für dieses Tutorial erstellen wir einen Cluster mit einer `t2.micro`-Container-Instance unter Verwendung des Amazon-ECS-optimierten Amazon Linux 2-AMI.

Informationen zum Erstellen eines Clusters für den EC2-Starttyp finden Sie unter the section called “Einen Cluster für den Amazon EC2 EC2-Starttyp erstellen”.

Schritt 5: Ausführen einer Amazon-ECS-Aufgabe

Sie können die Amazon-ECS-Konsole verwenden, um eine Aufgabe mithilfe der Aufgabendefinition auszuführen, die Sie erstellt haben. In diesem Tutorial führen wir eine Aufgabe mit dem EC2-Starttyp aus und verwenden dazu den Cluster, den wir im vorherigen Schritt erstellt haben.

Informationen zum Ausführen einer Aufgabe finden Sie unter the section called “Eine Anwendung als Aufgabe ausführen”.

Schritt 6: Überprüfen

Anhand folgender Schritte können Sie überprüfen, ob alle Schritte erfolgreich abgeschlossen wurden und die Umgebungsvariable in Ihrem Container ordnungsgemäß erstellt wurde.

Überprüfen, ob die Umgebungsvariable erstellt wurde

1. Suchen Sie nach der öffentlichen IP-Adresse oder DNS-Adresse für Ihre Container-Instance.
   a. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
   b. Wählen Sie im Navigationsbereich Cluster und dann den von Ihnen erstellten Cluster aus.
   c. Wählen Sie Infrastruktur und dann die Container-Instance aus.
   d. Zeichnen Sie die öffentliche IP oder das öffentliche DNS für Ihre Instance auf.
2. Stellen Sie auf einem macOS- oder Linux-Computer mit dem folgenden Befehl eine Verbindung mit Ihrer Instance her und ersetzen Sie den Pfad zu Ihrem privaten Schlüssel und die öffentliche Adresse für Ihre Instance:

   ```
   $ ssh -i /path/to/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com
   ```

   Weitere Informationen zur Verwendung eines Windows-Computers finden Sie unter Herstellen einer Verbindung zu Ihrer Linux-Instance von Windows aus mithilfe von PuTTY im Amazon EC2 EC2-Benutzerhandbuch.

   > **Important**
   >
   > Weitere Informationen zu Problemen beim Herstellen einer Verbindung zu Ihrer Instance finden Sie unter Troubleshooting Connecting to Your Instance im Amazon EC2 EC2-Benutzerhandbuch.

3. Erstellen Sie eine Liste der Container, die auf der Instance ausgeführt werden. Notieren Sie die Container-ID für `ecs-secrets-tutorial`-Container.

   ```
   docker ps
   ```

4. Stellen Sie mithilfe der Container-ID aus der Ausgabe des vorherigen Schritts eine Verbindung mit dem `ecs-secrets-tutorial`-Container her.

```
docker exec -it container_ID /bin/bash
```

5. Verwenden Sie den `echo`-Befehl, um den Wert der Umgebungsvariable zu drucken.

   ```
   echo $username_value
   ```

   Wenn das Tutorial erfolgreich war, sollten Sie die folgende Meldung sehen:

   ```
   password_value
   ```

   > **Note**
   >
   > Alternativ können Sie alle Umgebungsvariablen in Ihrem Container mithilfe des Befehls `env` (oder `printenv`) auflisten.

Schritt 7: Bereinigen

Wenn Sie mit diesem Tutorial fertig sind, sollten Sie die zugehörigen Ressourcen bereinigen, um zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen.

So bereinigen Sie die Ressourcen

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie Delete Cluster (Cluster löschen) aus.
5. Geben Sie im Bestätigungsfeld ***`Cluster-Name löschen`*** ein und wählen Sie dann Löschen.
6. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
7. Wählen Sie im Navigationsbereich Rollen aus.
8. Durchsuchen Sie die Liste der Rollen für `ecsTaskExecutionRole` und wählen Sie diese aus.
9. Wählen Sie Permissions und anschließend das X neben ECS SecretsTutorial aus. Wählen Sie Remove (Entfernen) aus.
10. Öffnen Sie die Secrets-Manager-Konsole unterhttps://console.aws.amazon.com/secretsmanager/.

11. Wählen Sie das von Ihnen erstellte username_value-Secret und Actions (Aktionen), Delete secret (Secret löschen).

Amazon ECS-Aufgabendefinitionsparameter

Aufgabendefinitionen sind in verschiedene Teile unterteilt: die Aufgabenfamilie, die AWS Identity and Access Management (IAM-) Aufgabenrolle, den Netzwerkmodus, Containerdefinitionen, Volumes, Einschränkungen bei der Aufgabenplatzierung und Starttypen. Die Familien- und Container-Definitionen sind in einer Aufgabendefinition obligatorisch. Im Gegensatz dazu sind Aufgabenrolle, Netzwerkmodus, Volumes, Aufgabenplatzierungseinschränkungen und Starttyp optional.

Sie können diese Parameter in einer JSON-Datei verwenden, um Ihre Aufgabendefinition zu konfigurieren.

Im Folgenden finden Sie weitere detaillierte Beschreibungen die einzelnen Aufgabendefinitionsparameter.

Familie

`family`

Typ: Zeichenfolge

Erforderlich: Ja

Wenn Sie eine Aufgabendefinition registrieren, vergeben Sie eine Familie, ähnlich einem Namen für mehrere Versionen der Aufgabendefinition, die mit einer Revisionsnummer angegeben ist. Die erste Aufgabendefinition, die in einer bestimmten Familie registriert wird, erhält die Revision 1 und alle danach registrierten Definitionen erhalten eine sequenzielle Revisionsnummer.

Starttypen

Wenn Sie eine Aufgabendefinition registrieren, können Sie einen Starttyp angeben, an dem Amazon ECS die Aufgabendefinition überprüfen soll. Eine Client-Ausnahme wird zurückgegeben, wenn die Aufgabendefinition nicht anhand der angegebenen Kompatibilitäten validiert. Weitere Informationen finden Sie unter Amazon-ECS-Starttypen.

Der folgende Parameter ist in einer Aufgabendefinition zulässig.

`requiresCompatibilities`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Zulässige Werte: `EC2` | `FARGATE` | `EXTERNAL`

Der Starttyp, der die Aufgabendefinition validiert. Dadurch kann überprüft werden, ob alle in der Aufgabendefinition verwendeten Parameter den Anforderungen des Starttyps entsprechen.

Aufgabenrolle

`taskRoleArn`

Typ: Zeichenfolge

Erforderlich: Nein

Wenn Sie eine Aufgabedefinition registrieren, können Sie eine Aufgabenrolle für eine IAM-Rolle bereitstellen, über die die Container in der Aufgabe die AWS -APIs aufrufen können, die in den zugehörigen Richtlinien in Ihrem Namen festgelegt sind. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

IAM-Rollen für Aufgaben unter Windows erfordern, dass die Option `-EnableTaskIAMRole` beim Starten des Amazon-ECS-optimierten Windows Server-AMI festgelegt ist. Ihre Container müssen außerdem bestimmten Konfigurationscode ausführen, damit dieses Feature verwendet werden kann. Weitere Informationen finden Sie unter Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance.

Aufgabenausführungsrolle

`executionRoleArn`

Typ: Zeichenfolge

Required: Conditional

Der Amazon-Ressourcenname (ARN) der Aufgabenausführungsrolle, die dem Amazon ECS-Container-Agenten die Erlaubnis erteilt, AWS API-Aufrufe in Ihrem Namen durchzuführen.

> **Note**
>
> Die IAM-Rolle für die Aufgabenausführung ist je nach den Anforderungen Ihrer Aufgabe erforderlich. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Netzwerkmodus

`networkMode`

Typ: Zeichenfolge

Erforderlich: Nein

Der Docker-Netzwerkmodus, der für die Container in der Aufgabe verwendet werden soll. Für Amazon-ECS-Aufgaben, die auf Amazon-EC2-Linux-Instances gehostet werden, sind die gültigen Werte `none`, `bridge`, `awsvpc` und `host`. Wenn kein Netzwerkmodus angegeben ist, ist der Standardnetzwerkmodus `bridge`. Für Amazon-ECS-Aufgaben, die auf Amazon EC2 Windows-Instances gehostet werden, sind die gültigen Werte `default` und`awsvpc`. Wenn kein Netzwerkmodus angegeben wird, wird der Netzwerkmodus `default` benutzt. Für auf Fargate gehostete Amazon ECS-Aufgaben ist der `awsvpc` Netzwerkmodus erforderlich.

Wenn der Netzwerkmodus auf `none` festgelegt ist, verfügen die Container der Aufgabe über keine externe Konnektivität und Port-Zuweisungen können nicht in der Container-Definition angegeben werden.

Im Netzwerkmodus `bridge` verwendet die Aufgabe das integrierte virtuelle Netzwerk von Docker unter Linux, das innerhalb jeder Amazon EC2-Instance ausgeführt wird, die die Aufgabe hostet. Das integrierte virtuelle Netzwerk unter Linux verwendet den `bridge` Docker-Netzwerktreiber.

Im Netzwerkmodus `host` verwendet die Aufgabe das Hostnetzwerk, das das integrierte virtuelle Netzwerk von Docker umgeht, indem es Container-Ports direkt der ENI der Amazon EC2-Instance zuweist, die die Aufgabe hostet. Dynamische Portzuordnungen können in diesem Netzwerkmodus nicht verwendet werden. Ein Container in einer Aufgabendefinition, der diesen Modus verwendet, muss eine bestimmte `hostPort`-Nummer angeben. Eine Portnummer auf einem Host kann nicht für mehrere Aufgaben verwendet werden. Daher können Sie nicht mehrere Aufgaben derselben Aufgabendefinition auf einer einzelnen Amazon EC2-Instance ausführen.

> **Important**
>
> Wenn Sie Aufgaben im Netzwerkmodus `host` ausführen, sollten Sie aus Sicherheitsgründen Container nicht über den Root-Benutzer (UID 0) ausführen. Verwenden Sie als bewährte Sicherheitsmethode immer einen Nicht-Root-Benutzer.

Wenn der Netzwerkmodus für die Amazon EC2 EC2-Starttypen lautet`awsvpc`, wird der Aufgabe eine elastic network interface zugewiesen, und Sie müssen eine angeben, `NetworkConfiguration` wenn Sie einen Service erstellen oder eine Aufgabe mit der Aufgabendefinition ausführen. Weitere Informationen finden Sie unter Netzwerkoptionen für Amazon ECS-Aufgaben für den EC2-Starttyp.

Im Netzwerkmodus `default` verwendet die Aufgabe das integrierte virtuelle Netzwerk von Docker unter Windows, das innerhalb jeder Amazon EC2-Instance ausgeführt wird, die die Aufgabe hostet. Das integrierte virtuelle Netzwerk unter Windows verwendet den `nat` Docker-Netzwerktreiber.

Wenn der Netzwerkmodus für die Fargate-Starttypen aktiviert ist`awsvpc`, wird der Aufgabe eine elastic network interface zugewiesen, und Sie müssen eine angeben, `NetworkConfiguration` wenn Sie einen Service erstellen oder eine Aufgabe mit der Aufgabendefinition ausführen. Weitere Informationen finden Sie unter Fargate Task Networking. Der Netzwerkmodus `awsvpc` bietet die höchste Netzwerkleistung für Container, da er den Amazon EC2-Netzwerkstapel verwendet. Offene Container-Ports werden direkt auf dem angeschlossenen Elastic-Network-Schnittstellenport abgebildet. Aus diesem Grund können Sie keine dynamischen Host-Port-Zuordnungen nutzen.

Die Netzwerkmodi `host` und `awsvpc` bieten die höchste Netzwerkleistung für Container, da sie den Amazon EC2-Netzwerkstapel verwenden. Für die Netzwerkmodi `host` und `awsvpc` sind die freigegebenen Container-Ports direkt auf den entsprechenden Host-Port (für den Netzwerkmodus `host`) oder den angehängten Elastic-Network-Schnittstellenanschluss (für den Netzwerkmodus `awsvpc`) abgebildet. Aus diesem Grund können Sie keine dynamischen Host-Port-Zuordnungen nutzen.

Wenn Sie den Fargate-Starttyp verwenden, ist der Netzwerkmodus `awsvpc` erforderlich. Wenn Sie den Starttyp EC2 verwenden, hängt der zulässige Netzwerkmodus vom zugrunde liegenden EC2-Instance-Betriebssystem ab. Unter Linux können beliebige Netzwerkmodi verwendet werden. Falls Windows, können Modi `default` und `awsvpc` verwendet werden.

Laufzeit-Plattform

`operatingSystemFamily`

Typ: Zeichenfolge

Required: Conditional

Standard: LINUX

Dieser Parameter ist für Amazon-ECS-Aufgaben erforderlich, die auf Fargate gehostet werden.

Wenn Sie eine Aufgabendefinition anmelden, geben Sie die Betriebssystemfamilie an.

Die gültigen Werte für auf Fargate gehostete Amazon-ECS-Aufgaben sind `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` und `WINDOWS_SERVER_2022_CORE`.

Die gültigen Werte für auf EC2 gehostete Amazon-ECS-Aufgaben sind `LINUX`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2019_FULL` und `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2016_FULL`, `WINDOWS_SERVER_2004_CORE` und `WINDOWS_SERVER_20H2_CORE`.

Alle Aufgabendefinitionen, die in einem Service verwendet werden, müssen den gleichen Wert für diesen Parameter aufweisen.

Wenn eine Aufgabendefinition Teil eines Services ist, muss dieser Wert mit dem `platformFamily`-Wert des Services übereinstimmen.

`cpuArchitecture`

Typ: Zeichenfolge

Required: Conditional

Standard: X86_64

Dieser Parameter ist für Amazon-ECS-Aufgaben erforderlich, die auf Fargate gehostet werden. Wenn der Parameter unverändert `null` bleibt, wird der Standardwert bei der Initiierung einer auf Fargate gehosteten Aufgabe automatisch zugewiesen.

Wenn Sie eine Aufgabendefinition anmelden, geben Sie die CPU-Architektur an. Die gültigen Werte sind `X86_64` und `ARM64`.

Alle Aufgabendefinitionen, die in einem Service verwendet werden, müssen den gleichen Wert für diesen Parameter aufweisen.

Wenn Sie Linux-Aufgaben für den Fargate-Launchtyp oder den EC2-Launchtyp haben, können Sie den Wert auf `ARM64` setzen. Weitere Informationen finden Sie unter the section called “Aufgabendefinitionen für 64-Bit-ARM-Workloads”.

Aufgabengröße

Wenn Sie eine Aufgabendefinition registrieren, können Sie den gesamten CPU- und Arbeitsspeicher für die Aufgabe angeben. Dies erfolgt separat von den Werten `cpu` und `memory` bei der Containerdefinition. Für Aufgaben, die auf Amazon-EC2-Instances gehostet werden, sind diese Felder optional. Für Aufgaben, die auf Fargate (Linux und Windows) gehostet werden, sind diese Felder erforderlich und es gibt für spezifische Werte für `cpu` und `memory`, die unterstützt werden.

> **Note**
>
> CPU- und Speicherparameter auf Aufgabenebene werden für Windows-Container ignoriert. Es wird empfohlen, für Windows-Container Ressourcen auf Container-Ebene festzulegen.

Der folgende Parameter ist in einer Aufgabendefinition zulässig:

`cpu`

Typ: Zeichenfolge

Required: Conditional

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

Das harte Limit der CPU-Einheiten, die für die Aufgabe vorhanden sind. Sie können CPU-Werte in der JSON-Datei als Zeichenfolge in CPU-Einheiten oder virtuellen CPUs (vCPUs) angeben.

Sie können beispielsweise einen CPU-Wert entweder in CPU-Einheiten oder `1024` `1 vCPU` in vCPUs angeben. Wenn die Aufgabendefinition registriert ist, wird ein vCPU-Wert in eine Ganzzahl umgewandelt, die die CPU-Einheiten angibt.

Für Aufgaben, die auf externen oder Amazon-EC2-Instances ausgeführt werden, ist dieses Feld optional. Wenn Ihr Cluster keine registrierten Container-Instances mit den entsprechenden CPU-Einheiten hat, schlägt die Aufgabe fehl. Unterstützte Werte für Aufgaben, die auf EC2 oder externen Instances ausgeführt werden, liegen zwischen `0.125` vCPUs und `10` vCPUs.

Für Aufgaben, die auf Fargate ausgeführt werden (sowohl Linux- als auch Windows-Container), ist dieses Feld erforderlich. Sie müssen einen der folgenden Werte verwenden, der Ihren Bereich der unterstützten Werte für den `memory`-Parameter festlegt. Die folgende Tabelle zeigt die gültigen Kombinationen von CPU- und Speicherauslastung auf Aufgabenebene.

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Zwischen 4 GB und 16 GB in 1-GB-Schritten	Linux, Windows
4096 (4 vCPU)	Zwischen 8 GB und 30 GB in 1-GB-Schritten	Linux, Windows
8 192 (8 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 16 GB und 60 GB in 4-GB-Schritten	Linux

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
16 384 (16 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 32 GB und 120 GB in 8-GB-Schritten	Linux

`memory`

Typ: Zeichenfolge

Required: Conditional

Note

Dieser Parameter wird für Windows-Container nicht unterstützt.

Die feste Speichergrenze, die der Aufgabe zugewiesen werden soll. Sie können Speicherwerte in der Aufgabendefinition als Zeichenfolge in Mebibyte (MiB) oder Gigabyte (GB) angeben. Sie können beispielsweise einen Speicherwert entweder `3072` in MiB oder `3 GB` in GB angeben. Wenn die Aufgabendefinition registriert ist, wird ein GB-Wert in eine Ganzzahl umgewandelt, die die MiB angibt.

Für Aufgaben, die auf Amazon-EC2-Instances gehostet werden, ist dieses Feld optional und jeder Wert kann verwendet werden. Wenn ein Speicherwert auf Aufgabenebene angegeben wird, ist der Speicherwert auf Container-Ebene optional. Wenn Ihr Cluster keine registrierten Container-Instances mit dem erforderlichen Arbeitsspeicher hat, schlägt die Aufgabe fehl. Sie können Ihre Ressourcennutzung maximieren, indem Sie Ihren Aufgaben so viel Arbeitsspeicher wie möglich für einen bestimmten Instance-Typ zuweisen. Weitere Informationen finden Sie unter Reservieren von Amazon ECS Linux-Container-Instance-Speicher .

Für Aufgaben, die auf Fargate gehostet werden (sowohl Linux- als auch Windows-Container), ist dieses Feld erforderlich, und Sie müssen einen der folgenden Werte verwenden, der den Bereich der unterstützten Werte für den `cpu`-Parameter bestimmt:

Speicherwert (in MiB, mit ungefährem Äquivalenzwert in GB)	CPU-Wert	Für Fargate unterstützte Betriebssysteme
512 (0.5 GB), 1024 (1 GB), 2048 (2 GB)	256 (0,25 vCPU)	Linux
1024 (1 GB), 2048 (2 GB), 3072 (3 GB), 4096 (4 GB)	512 (0,5 vCPU)	Linux
2048 (2 GB), 3072 (3 GB), 4096 (4GB), 5120 (5 GB), 6144 (6 GB), 7168 (7 GB), 8192 (8 GB)	1024 (1 vCPU)	Linux, Windows
Zwischen 4096 (4 GB) und 16384 (16 GB) in Schritten von 1024 (1 GB)	2048 (2 vCPU)	Linux, Windows
Zwischen 8192 (8 GB) und 30720 (30 GB) in Schritten von 1024 (1 GB)	4096 (4 vCPU)	Linux, Windows
Zwischen 16 GB und 60 GB in 4-GB-Schritten **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	8 192 (8 vCPU)	Linux

Speicherwert (in MiB, mit ungefährem Äquivalenzwert in GB)	CPU-Wert	Für Fargate unterstützte Betriebssysteme
Zwischen 32 GB und 120 GB in 8-GB-Schritten **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	16 384 (16 vCPU)	Linux

Containerdefinitionen

Wenn Sie eine Aufgabendefinition registrieren, müssen Sie eine Liste der Containerdefinitionen angeben, die an den Docker-Daemon auf einer Container-Instance übergeben werden. Die folgenden Parameter sind in einer Containerdefinition zulässig.

Themen

- Standardparameter für Containerdefinition
- Erweiterte Parameter für Containerdefinitionen
- Andere Parameter für die Containerdefinition

Standardparameter für Containerdefinition

Die folgenden Aufgabendefinitionsparameter sind in Containerdefinitionen entweder erforderlich oder werden meistens verwendet.

Themen

- Name
- Image
- Arbeitsspeicher
- Port-Zuweisungen

- Private-Repository-Daten

Name

`name`

Typ: Zeichenfolge

Erforderlich: Ja

Der Name eines Containers. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig. Wenn Sie mehrere Container in einer Aufgabedefinition verknüpfen, kann der `name` eines Containers in die `links` eines anderen Containers eingefügt werden. Das dient dazu, die Container zu verbinden.

Image

`image`

Typ: Zeichenfolge

Erforderlich: Ja

Das Image zum Starten eines Containers. Diese Zeichenfolge wird direkt an den Docker-Daemon übergeben. Images in der Docker Hub-Registrierung sind standardmäßig verfügbar. Sie können entweder mit *`repository-url`*`/`*`image`*`:`*`tag`* oder mit *`repository-url`*`/`*`image`*`@`*`digest`* auch andere Repositorys angeben. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche, Unterstriche, Doppelpunkte, Punkte und Schrägstriche sind zulässig. Dieser Parameter wird `Image` zugeordnet im Abschnitt Create a container (Erstellen eines Containers) im Docker-Remote-API und dem `IMAGE`-Parameter von docker run.

- Wenn eine neue Aufgabe gestartet wird, ruft der Amazon-ECS-Container-Agent die neueste Version des angegebenen Image und das Tag für den Container ab, der verwendet werden soll. Allerdings werden nachfolgende Aktualisierungen eines Repository-Images nicht an Aufgaben übertragen, die bereits ausgeführt werden.
- Images in privaten Registrierungen werden unterstützt. Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.
- Images in Amazon ECR-Repositorys können entweder mit der vollständigen `registry/repository:tag`- oder der `registry/repository@digest`-Namenskonvention angegeben werden (beispielsweise

aws_account_id.dkr.ecr.*region*.amazonaws.com/*my-web-app*:*latest* oder *aws_account_id*.dkr.ecr.*region*.amazonaws.com/*my-web-app*@*sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE*).

- Images in offiziellen Repositorys in Docker Hub verwenden einen einzelnen Namen (z. B. `ubuntu` oder `mongo`).
- Images in anderen Repositorys in Docker Hub sind mit einem Organisationsnamen qualifiziert (z. B `amazon/amazon-ecs-agent`).
- Image in anderen Online-Repositorys sind durch einen Domain-Namen zusätzlich qualifiziert (z. B. `quay.io/assemblyline/ubuntu`).

Arbeitsspeicher

`memory`

Typ: Ganzzahl

Erforderlich: Nein

Die Menge des Arbeitsspeichers (in MiB), die dem Container bereitgestellt wird. Wenn Ihr Container versucht, das hier angegebene Limit zu überschreiten, wird der Container beendet. Die gesamte Speicherkapazität, die für alle Container reserviert ist, muss niedriger sein als der Aufgabenwert `memory`, sofern angegeben. Dieser Parameter ordnet zu `Memory` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory` für die docker run zu.

Wenn Sie den Fargate-Starttyp verwenden, ist dieser Parameter optional.

Wenn Sie den EC2-Starttyp verwenden, müssen Sie entweder einen Speicherwert auf Aufgabenebene oder einen Speicherwert auf Container-Ebene angeben. Wenn Sie sowohl einen Wert auf Containerebene `memory` und `memoryReservation` angeben, muss der Wert `memory` größer sein als der Wert `memoryReservation`. Wenn Sie `memoryReservation` angeben, wird dieser Wert von den verfügbaren Speicherressourcen für die Container-Instance abgezogen, auf der der Container platziert ist. Andernfalls wird der Wert `memory` verwendet.

Der Docker-Daemon 20.10.0 oder neuer reserviert mindestens 6 MiB Arbeitsspeicher für einen Container. Geben Sie daher nicht weniger als 6 MiB Arbeitsspeicher für Ihre Container an.

Der Docker-Daemon 19.03.13-ce oder älter reserviert mindestens 4 MiB Arbeitsspeicher für einen Container. Geben Sie daher nicht weniger als 4 MiB Arbeitsspeicher für Ihre Container an.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Aufgaben so viel Arbeitsspeicher wie möglich für einen bestimmten Instance-Typ zuweisen, lesen Sie nach unter Reservieren von Amazon ECS Linux-Container-Instance-Speicher .

`memoryReservation`

Typ: Ganzzahl

Erforderlich: Nein

Die weiche Arbeitsspeichergrenze (in MiB) für die Reservierung für den Container. Wenn der Systemspeicher strittig ist, versucht Docker den Arbeitsspeicher des Containers innerhalb der weichen Grenze zu halten. Ihr Container kann jedoch bei Bedarf mehr Speicher verwenden. Der Container kann bis zu der mit dem Parameter `memory` angegebenen harten Grenze (falls zutreffend) oder bis zum gesamten verfügbaren Speicher der Container-Instance verwendet werden, je nachdem, was zuerst eintritt. Dieser Parameter ordnet zu `MemoryReservation` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--memory-reservation` für die docker run zu.

Wenn kein Speicherwert auf Aufgabenebene angegeben ist, müssen Sie eine Ganzzahl ungleich Null für einen oder beide Werte von `memory` oder `memoryReservation` in einer Container-Definition angeben. Wenn Sie beide angeben, muss `memory` größer als `memoryReservation` sein. Wenn Sie `memoryReservation` angeben, wird dieser Wert von den verfügbaren Speicherressourcen für die Container-Instance abgezogen, auf der der Container platziert ist. Andernfalls wird der Wert `memory` verwendet.

Nehmen wir zum Beispiel an, dass Ihr Container normalerweise 128 MiB Arbeitsspeicher verwendet, aber gelegentlich kurzzeitig auf 256 MiB Arbeitsspeicher expandiert. Sie können einen `memoryReservation` von 128 MiB und einen festen `memory`-Grenzwert von 300 MiB festlegen. Mit dieser Konfiguration kann der Container nur 128 MiB Arbeitsspeicher von den verbleibenden Ressourcen auf der Container-Instance reservieren. Gleichzeitig ermöglicht die Konfiguration auch, dass der Container bei Bedarf mehr Speicherressourcen verwenden kann.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

Der Docker-Daemon 20.10.0 oder neuer reserviert mindestens 6 MiB Arbeitsspeicher für einen Container. Geben Sie daher nicht weniger als 6 MiB Arbeitsspeicher für Ihre Container an.

Der Docker-Daemon 19.03.13-ce oder älter reserviert mindestens 4 MiB Arbeitsspeicher für einen Container. Geben Sie daher nicht weniger als 4 MiB Arbeitsspeicher für Ihre Container an.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Aufgaben so viel Arbeitsspeicher wie möglich für einen bestimmten Instance-Typ zuweisen, lesen Sie nach unter Reservieren von Amazon ECS Linux-Container-Instance-Speicher .

Port-Zuweisungen

`portMappings`

Typ: Objekt-Array

Erforderlich: Nein

Port-Zuweisungen ermöglichen Containern den Zugriff auf Ports der Host-Container-Instance, um Datenverkehr zu senden oder zu empfangen.

Legen Sie für Aufgabendefinitionen, die den Netzwerkmodus `awsvpc` verwenden, nur `containerPort` fest. Der `hostPort` kann leer bleiben oder den gleichen Wert wie der `containerPort` haben.

Die Port-Zuweisungen unter Windows verwenden die Gateway-Adresse `NetNAT` und nicht `localhost`. Für Port-Zuweisungen unter Windows gibt es kein Loopback, daher können Sie auch nicht vom Host selbst auf den zugewiesenen Port eines Containers zugreifen.

Die meisten Felder dieses Parameters (inklusive `containerPort`, `hostPort`, `protocol`) werden `PortBindings` im Abschnitt Container erstellen der Docker-Remote-API zugeordnet und die Option `--publish` wird docker run zugeordnet. Wenn der Netzwerkmodus einer Aufgabendefinition auf `host` festgelegt ist, müssen die Host-Ports entweder undefiniert sein oder mit dem Container-Port in der Port-Zuweisung übereinstimmen.

> **Note**
>
> Nachdem eine Aufgabe den Status `RUNNING` erreicht hat, sind manuelle und automatische Host- und Container-Port-Zuordnungen an den folgenden Orten sichtbar:
>
> - Konsole: Abschnitt Network Bindings (Netzwerk-Bindungen) einer Container-Beschreibung für eine bestimmte Aufgabe.
> - AWS CLI: Der Abschnitt `networkBindings` der Befehlsausgabe describe-tasks.
> - API: `DescribeTasks`-Antwort.
> - Metadaten: Der Endpunkt der Aufgabenmetadaten.

`appProtocol`

Typ: Zeichenfolge

Erforderlich: Nein

Das Anwendungsprotokoll, das für die Portzuordnung verwendet wird. Dieser Parameter gilt nur für Service Connect. Wir empfehlen Ihnen, diesen Parameter so festzulegen, dass er mit dem von Ihrer Anwendung verwendeten Protokoll konsistent ist. Wenn Sie diesen Parameter festlegen, fügt Amazon ECS dem Service-Connect-Proxy eine protokollspezifische Verbindungsbehandlung hinzu. Wenn Sie diesen Parameter festlegen, fügt Amazon ECS protokollspezifische Telemetrie in der Amazon ECS-Konsole und hinzu. CloudWatch

Wenn Sie keinen Wert für diesen Parameter festlegen, wird TCP verwendet. Amazon ECS fügt jedoch keine protokollspezifische Telemetrie für TCP hinzu.

Weitere Informationen finden Sie unter the section called “ Service Connect”.

Gültige Protokollwerte: `"HTTP" | "HTTP2" | "GRPC"`

`containerPort`

Typ: Ganzzahl

Erforderlich: Ja, wenn `portMappings` verwendet werden

Die Port-Nummer auf dem Container, der an den vom Benutzer angegebenen oder automatisch zugewiesenen Host-Port gebunden ist.

Wenn Sie Container in einer Aufgabe mit dem Starttyp Fargate verwenden, müssen offengelegte Ports mit `containerPort` spezifiziert werden.

Für Windows-Container auf Fargate können Sie Port 3150 nicht als `containerPort` verwenden. Das liegt daran, dass er reserviert ist.

Angenommen, Sie verwenden Container in einer Aufgabe mit dem Starttyp EC2 und geben einen Container-Port und keinen Host-Port an. Dann erhält Ihr Container automatisch einen Host-Port im flüchtigen Port-Bereich. Weitere Informationen finden Sie unter `hostPort`. Port-Zuweisungen, die auf diese Weise automatisch zugewiesen werden, werden beim Kontingent der 100 reservierten Ports für eine Container-Instance nicht mitgezählt.

`containerPortRange`

Typ: Zeichenfolge

Erforderlich: Nein

Der Portnummernbereich des Containers, der an den dynamisch zugeordneten Host-Portbereich gebunden ist.

Sie können diesen Parameter nur mithilfe der `register-task-definition`-API festlegen. Die Option ist im `portMappings`-Parameter verfügbar. Weitere Informationen finden Sie unter register-task-definition in der AWS Command Line Interface -Referenz.

Die folgenden Regeln gelten, wenn Sie ein `containerPortRange` angeben :

- Sie müssen entweder den `bridge`-Netzwerkmodus oder den `awsvpc`-Netzwerkmodus verwenden.
- Dieser Parameter ist sowohl für den EC2- als auch für den AWS -Fargate-Starttyp verfügbar.
- Dieser Parameter ist sowohl für Windows- als auch für Linux-Betriebssysteme verfügbar.
- Die Container-Instance muss mindestens über Version 1.67.0 des Container-Agenten und mindestens über Version 1.67.0-1 des `ecs-init`-Pakets verfügen.
- Sie können bis zu 100 Portbereiche für jeden Container angeben.
- Sie geben keine `hostPortRange` an. Der Wert von `hostPortRange` ist wie folgt festgelegt:
  - Für Container in einer Aufgabe mit dem `awsvpc`-Netzwerkmodus wird `hostPort` auf denselben Wert wie `containerPort` festgelegt. Dies ist eine statische Zuordnungsstrategie.

- Für Container in einer Aufgabe mit dem `bridge`-Netzwerkmodus findet der Amazon-ECS-Agent offene Host-Ports aus dem flüchtigen Standardbereich und übergibt sie an Docker, um sie an die Container-Ports zu binden.

- Gültige Werte für `containerPortRange` liegen zwischen 1 and 65 535.
- Ein Port kann nur in einer Portzuordnung für jeden Container enthalten sein.
- Sie können keine überlappenden Portbereiche angeben.
- Die erste Port im Bereich muss kleiner als der letzte Port im Bereich sein.
- Docker empfiehlt, den Docker-Proxy in der Konfigurationsdatei des Docker-Daemons zu deaktivieren, wenn Sie eine große Anzahl von Ports haben.

  Weitere Informationen finden Sie in Ausgabe #11185 unter. GitHub

  Informationen zum Deaktivieren des Docker-Proxy in der Docker-Daemon-Konfigurationsdatei finden Sie unter Docker-Daemon im Amazon-ECS-Entwicklerhandbuch.

Sie können `DescribeTasks` aufrufen, um die `hostPortRange` anzuzeigen, bei denen es sich um die Host-Ports handelt, die an die Container-Ports gebunden sind.

Die Portbereiche sind nicht in den Amazon ECS-Aufgabenereignissen enthalten, die an gesendet werden EventBridge. Weitere Informationen finden Sie unter the section called “Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge”.

`hostPortRange`

Typ: Zeichenfolge

Erforderlich: Nein

Der Portnummernbereich auf dem Host, der mit der Netzwerkbindung verwendet wird. Dieser wird von Docker zugewiesen und vom Amazon-ECS-Agenten bereitgestellt.

`hostPort`

Typ: Ganzzahl

Erforderlich: Nein

Die Port-Nummer auf der Container-Instance, die für Ihren Container reserviert werden soll.

Wenn Sie Container in einer Aufgabe mit dem Starttyp Fargate verwenden, kann `hostPort` entweder leer bleiben oder muss denselben Wert wie `containerPort` haben.

Angenommen, Sie verwenden Container in einer Aufgabe mit dem Starttyp EC2. Sie können einen nicht reservierten Host-Port für Ihre Container-Port-Zuordnung angeben. Dies wird als statische Host-Port-Zuordnung bezeichnet. Sie können den `hostPort` auch weglassen (oder auf `0` setzen), während Sie einen `containerPort` angeben. Ihr Container erhält automatisch einen Port im kurzlebigen Portbereich für Ihr Container-Instance-Betriebssystem und Ihre Docker-Version. Dies wird als dynamische Host-Port-Zuordnung bezeichnet.

Der standardmäßige flüchtige Port-Bereich für Docker-Version 1.6.0 und später wird in der Instance unter `/proc/sys/net/ipv4/ip_local_port_range` aufgelistet. Wenn dieser Kernel-Parameter nicht verfügbar ist, wird der standardmäßige flüchtige Port-Bereich von `49153–65535` verwendet. Versuchen Sie nicht, einen Host-Port im flüchtigen Portbereich anzugeben. Das liegt daran, dass diese für die automatische Zuweisung reserviert sind. Im Allgemeinen zählen Ports unter `32768` nicht zum flüchtigen Port-Bereich.

Die standardmäßigen reservierten Ports sind `22` für SSH, die Docker-Ports, `2375` und `2376` und der Amazon-ECS-Container-Agenten-Port `51678-51680`. Jeder Host-Port, der zuvor vom Benutzer für eine laufende Aufgabe festgelegt wurde, ist auch während der Ausführung der Aufgabe reserviert. Nach dem Beenden einer Aufgabe wird der Host-Port freigegeben. Die aktuell reservierten Ports werden in der `remainingResources` der describe-container-instances-Ausgabe angezeigt. Eine Container-Instance kann bis zu 100 reservierte Ports gleichzeitig haben, einschließlich der standardmäßig reservierten Ports. Automatisch zugewiesene Ports werden nicht auf das Kontingent von 100 reservierten Ports angerechnet.

`name`

Typ: Zeichenfolge

Erforderlich: Nein, erforderlich für die Konfiguration von Service Connect in einem Service

Der Name, der für die Portzuordnung verwendet wird. Dieser Parameter gilt nur für Service Connect. Dieser Parameter ist der Name, den Sie in der Service-Connect-Konfiguration eines Services verwenden.

Weitere Informationen finden Sie unter Verwenden Sie Service Connect, um Amazon ECS-Services mit Kurznamen zu verbinden.

Im folgenden Beispiel werden beide erforderlichen Felder für Service Connect verwendet.

```
"portMappings": [
```

```
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Typ: Zeichenfolge

Erforderlich: Nein

Das für die Port-Zuweisung verwendete Protokoll. Gültige Werte sind `tcp` und `udp`. Der Standardwert ist `tcp`.

> **Important**
>
> Nur `tcp` wird für Service Connect unterstützt. Denken Sie daran, dass `tcp` impliziert ist, wenn dieses Feld nicht festgelegt ist.

> **Important**
>
> UDP-Unterstützung ist nur auf Container-Instances verfügbar, die mit Version 1.2.0 des Amazon-ECS-Container-Agenten (z. B. `amzn-ami-2015.03.c-amazon-ecs-optimized`-AMI) oder höher gestartet wurden – oder mit Container-Agenten, die auf Version 1.3.0 oder höher aktualisiert wurden. Die neueste Version für Ihren Container-Agenten finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Verwenden Sie die folgende Syntax, um einen Host-Port anzugeben.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Verwenden Sie die folgende Syntax, um einen Host-Port automatisch zuzuweisen.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Private-Repository-Daten

`repositoryCredentials`

Typ: RepositoryCredentials Objekt

Erforderlich: Nein

Die Repository-Anmeldeinformationen für die Authentifizierung bei privaten Registrierungen.

Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.

`credentialsParameter`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `repositoryCredentials` verwendet werden

Der Amazon-Ressourcenname (ARN) des Secrets mit den Anmeldeinformationen für das private Repository.

Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.

> **Note**
>
> Wenn Sie die Amazon ECS-API oder AWS SDKs verwenden und der geheime Schlüssel in derselben Region wie die Aufgabe existiert, die Sie starten, können Sie entweder den vollständigen ARN oder den Namen des Geheimnisses verwenden. AWS CLI Wenn Sie den verwenden AWS Management Console, müssen Sie den vollständigen ARN des Geheimnisses angeben.

Im Folgenden finden Sie einen Ausschnitt einer Aufgabendefinition, welche die erforderlichen Parameter zeigt:

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Erweiterte Parameter für Containerdefinitionen

Die folgenden erweiterten Parameter für Container-Definitionen bieten erweiterte Funktionen für den docker run-Befehl , der verwendet wird, um Container auf Ihren Amazon-ECS-Container-Instances zu starten.

Themen

- Zustandsprüfung
- Umgebung
- Network settings (Netzwerkeinstellungen)
- Speicher und Protokollierung
- Sicherheit
- Ressourcenlimits
- Docker-Bezeichnungen

Zustandsprüfung

`healthCheck`

Der Befehl für die Container-Zustandsprüfung und die zugehörigen Konfigurationsparameter für den Container. Weitere Informationen finden Sie unter Ermitteln Sie den Zustand von Amazon ECS-Aufgaben mithilfe von Container-Zustandsprüfungen.

`command`

Ein Zeichenfolgen-Array, das den Befehl darstellt, den der Container ausführt, um festzustellen, ob er fehlerfrei ist. Das Zeichenfolge-Array kann mit `CMD` beginnen, um die Befehlsargumente direkt auszuführen, oder mit `CMD-SHELL`, um den Befehl mit der Standard-Shell des Containers auszuführen. Ist nichts davon angegeben, wird `CMD` verwendet.

Wenn Sie eine Aufgabendefinition in der registrieren AWS Management Console, verwenden Sie eine durch Kommas getrennte Liste von Befehlen. Diese Befehle werden in eine Zeichenfolge konvertiert, nachdem die Aufgabendefinition erstellt wurde. Im Folgenden finden Sie eine Beispieleingabe für eine Zustandsprüfung.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Wenn Sie eine Aufgabendefinition mithilfe des AWS Management Console JSON-Bereichs, der AWS CLI oder der APIs registrieren, schließen Sie die Befehlsliste in Klammern ein. Im Folgenden finden Sie eine Beispieleingabe für eine Zustandsprüfung.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Ein Beendigungscode von 0 ohne `stderr`-Ausgabe zeigt einen Erfolg an, und der Beendigungscode ungleich Null zeigt einen Fehler an. Weitere Informationen finden Sie unter `HealthCheck` im Bereich Create a container der Docker Remote API.

`interval`

Der Zeitraum (in Sekunden) zwischen den Zustandsprüfungen. Sie können zwischen 5 und 300 Sekunden angeben. Der Standardwert ist 30 Sekunden.

`timeout`

Der Zeitraum (in Sekunden), der angibt, wie lange gewartet wird, bis eine Zustandsprüfung erfolgreich ist, bevor sie als fehlerhaft betrachtet wird. Sie können zwischen 2 und 60 Sekunden angeben. Der Standardwert liegt bei 5 Sekunden.

`retries`

Die Anzahl, wie oft eine fehlgeschlagene Zustandsprüfung wiederholt wird, bevor der Container als fehlerhaft betrachtet wird. Sie können zwischen 1 und 10 Wiederholungen angeben. Der Standardwert ist drei Wiederholungsversuche.

`startPeriod`

Der optionale Übergangszeitraum, der angibt, wie lange der Container Zeit für einen Bootstrap hat, bevor fehlgeschlagene Zustandsprüfungen der maximalen Anzahl an Wiederholungen angerechnet werden. Sie können zwischen 0 und 300 Sekunden angeben. Standardmäßig ist `startPeriod` deaktiviert.

Umgebung

`cpu`

Typ: Ganzzahl

Erforderlich: Nein

Die Anzahl der physischen `cpu`-Einheiten, die der Amazon-ECS-Container-Agent für den Container reserviert. In Linux ordnet dieser Parameter zu `CpuShares` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--cpu-shares` für die docker run zu.

Dieses Feld ist für Aufgaben mit dem Fargate-Starttyp optional. Die Gesamtmenge an CPU, die für alle Container innerhalb einer Aufgabe reserviert ist, muss niedriger sein als der `cpu`-Wert auf Aufgabenebene.

> **Note**
>
> Sie können die Anzahl der CPU-Einheiten bestimmen, die für jeden Amazon-EC2-Instance-Typ zur Verfügung stehen. Multiplizieren Sie dazu die Anzahl der vCPUs, die für den jeweiligen Instance-Typ auf der Detailseite Amazon-EC2-Instances aufgeführt sind, mit 1 024.

Linux-Container teilen sich nicht zugewiesene CPU-Einheiten mit anderen Containern auf der Container-Instance im gleichen Verhältnis wie die ihnen zugewiesene Menge. Nehmen Sie zum Beispiel an, dass Sie eine Aufgabe für einen einzelnen Container auf einer Instance mit einem Kern und 512 CPU-Einheiten für diesen Container ausführen. Außerdem ist diese Aufgabe die einzige Aufgabe, die auf der Container-Instance läuft. In diesem Beispiel kann der Container jederzeit die gesamte Menge von 1 024 CPU-Einheiten nutzen. Angenommen, Sie haben jedoch auf dieser Container-Instance eine weitere Kopie der gleichen Aufgabe gestartet. Jede Aufgabe

würde bei Bedarf eine garantierte Menge von mindestens 512 CPU-Einheiten erhalten. Ebenso kann jeder Container zu einer höheren CPU-Auslastung übergehen, wenn der andere Container die verbleibende CPU nicht nutzt. Wenn jedoch beide Aufgaben die ganze Zeit 100 % aktiv sind, stünden ihnen jeweils nur 512 CPU-Einheiten zur Verfügung.

Auf Linux-Container-Instances verwendet der Docker-Daemon auf der Container-Instance den CPU-Wert, um das relative CPU-Anteilsverhältnis für die laufenden Container zu berechnen. Weitere Informationen finden Sie unter CPU share constraint in der Docker-Dokumentation. Der kleinste gültige CPU-Freigabe wert, den der Linux-Kernel zulässt, ist 2. Der CPU-Parameter ist jedoch nicht erforderlich und Sie können CPU-Werte unter 2 in Ihren Container-Definitionen verwenden. Bei CPU-Werten unter 2 (einschließlich 0) variiert das Verhalten je nach Version Ihres Amazon-ECS-Container-Agenten:

- Agent-Versionen <= 1.1.0: Null-CPU-Werte werden an Docker als 0 übergeben, die von Docker dann in 1 024 CPU-Anteile konvertiert werden. CPU-Werte von 1 werden an Docker als 1 übergeben, die der Linux-Kernel in zwei CPU-Freigaben konvertiert.
- Agent-Versionen >= 1.2.0: Null-CPU-Werte und CPU-Werte von 1 werden an Docker als zwei CPU-Anteile übergeben.

Auf Windows-Container-Instances wird das CPU-Kontingent als absolutes Kontingent durchgesetzt. Windows-Container haben nur Zugriff auf die in der Aufgabendefinition festgelegte Menge an CPU. Ein Nullwert oder ein CPU-Wert von Null wird an Docker als `0` übergeben. Windows interpretiert diesen Wert dann als 1 % einer CPU.

Weitere Beispiele finden Sie unter So verwaltet Amazon ECS CPU- und Arbeitsspeicher-Ressourcen.

`gpu`

Typ: ResourceRequirement Objekt

Erforderlich: Nein

Die Anzahl der physischen `GPUs`, die der Amazon-ECS-Container-Agent für den Container reserviert. Die Anzahl der für alle Container in einer Aufgabe reservierten GPUs darf nicht größer sein als die Anzahl der verfügbaren GPUs für die Container-Instance, für die die Aufgabe gestartet wird. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.

> **Note**
>
> Dieser Parameter wird nicht für Windows-Container oder Container unterstützt, die auf Fargate gehostet werden.

`Elastic Inference accelerator`

Typ: ResourceRequirement Objekt

Erforderlich: Nein

Für den `InferenceAccelerator`-Typ stimmt der `value` mit dem `deviceName` für einen `InferenceAccelerator` überein, der in einer Aufgabendefinition angegeben ist. Weitere Informationen finden Sie unter the section called “Der Name des Elastic-Inference-Beschleunigers”.

> **Note**
>
> Ab 15. April 2023 AWS wird Amazon Elastic Inference (EI) keine Neukunden mehr in Amazon Elastic Inference (EI) einbinden und Bestandskunden dabei helfen, ihre Workloads auf Optionen umzustellen, die ein besseres Preis und eine bessere Leistung bieten. Nach dem 15. April 2023 können Neukunden keine Instances mit Amazon EI-Beschleunigern in Amazon SageMaker, Amazon ECS oder Amazon EC2 starten. Kunden, die Amazon EI in den letzten 30 Tagen mindestens einmal genutzt haben, gelten jedoch als aktuelle Kunden und können den Service weiterhin nutzen.

> **Note**
>
> Dieser Parameter wird nicht für Windows-Container oder Container unterstützt, die auf Fargate gehostet werden.

`essential`

Typ: Boolesch

Erforderlich: Nein

Nehmen wir an, der `essential`-Parameter eines Containers ist als `true` gekennzeichnet und dieser Container schlägt fehl oder wird aus irgendeinem Grund beendet. Dann werden alle anderen Container beendet, die Teil der Aufgabe sind. Wenn der Parameter `essential` eines Containers als `false` gekennzeichnet ist, wirkt sich ein Ausfall dieses Containers nicht auf die anderen Container in der Aufgabe aus. Wenn dieser Parameter ausgelassen wird, wird davon ausgegangen, dass ein Container „essential“ (entscheidend) ist.

Alle Aufgaben müssen über mindestens einen entscheidenden Container verfügen. Angenommen, Sie haben eine Anwendung, die aus mehreren Containern besteht. Dann gruppieren Sie Container, die für einen gemeinsamen Zweck verwendet werden, in Komponenten und teilen die verschiedenen Komponenten in mehrere Aufgabendefinitionen auf. Weitere Informationen finden Sie unter Entwickeln Sie Ihre Anwendung für Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Von frühen Versionen des Amazon-ECS-Container-Agenten werden die `entryPoint`-Parameter nicht korrekt verarbeitet. Wenn Sie Probleme bei der Verwendung von `entryPoint` haben, aktualisieren Sie Ihren Container-Agenten oder geben Sie Ihre Befehle und Argumente stattdessen als `command`-Array-Objekte an.

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Der Eintrittspunkt, der an den Container übergeben wird. Dieser Parameter ordnet zu `Entrypoint` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--entrypoint` für die docker run zu. Weitere Informationen zum Docker-Parameter `ENTRYPOINT` finden Sie unter https://docs.docker.com/engine/reference/builder/#entrypoint.

```
"entryPoint": ["string", ...]
```

`command`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Der Befehl, der an den Container übergeben wird. Dieser Parameter ist `Cmd` im Abschnitt Erstellen eines Containers der Docker Remote-API und dem Parameter `COMMAND` von docker run zugeordnet. Weitere Informationen zum Docker-Parameter `CMD` finden Sie unter https://docs.docker.com/engine/reference/builder/#cmd. Wenn mehrere Argumente vorhanden sind, stellen Sie sicher, dass jedes Argument eine getrennte Zeichenfolge im Array ist.

```
"command": ["string", ...]
```

`workingDirectory`

Typ: Zeichenfolge

Erforderlich: Nein

Das Arbeitsverzeichnis, in dem Befehle im Container ausgeführt werden sollen. Dieser Parameter ordnet zu `WorkingDir` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--workdir` für die docker run zu.

```
"workingDirectory": "string"
```

`environmentFiles`

Typ: Objekt-Array

Erforderlich: Nein

Eine Liste von Dateien, die die Umgebungsvariablen enthalten, die an einen Container übergeben werden sollen. Dieser Parameter wird auf die Option `--env-file` abgebildet, um auf die docker run zu verweisen.

Dies ist nicht für Windows Container und Windows-Container auf Fargate verfügbar

Sie können bis zu 10 Umgebungsdateien angeben. Die Datei muss eine `.env` Dateierweiterung haben. Jede Zeile in einer Umgebungsdatei enthält eine Umgebungsvariable im Format `VARIABLE=VALUE`. Zeilen, die mit # beginnen, werden als Kommentare behandelt und ignoriert. Weitere Informationen zur Syntax der entsprechenden Umgebungsvariablen finden Sie unter Declare default environment variables in file (Deklarieren von Standardumgebungsvariablen in Datei).

Wenn in der Containerdefinition einzelne Umgebungsvariablen angegeben sind, haben sie Vorrang vor den Variablen, die in einer Umgebungsdatei enthalten sind. Wenn mehrere Umgebungsdateien angegeben werden, die dieselbe Variable enthalten, werden sie von oben nach unten verarbeitet. Es wird empfohlen, eindeutige Variablennamen zu verwenden. Weitere Informationen finden Sie unter Übergeben Sie eine einzelne Umgebungsvariable an einen Amazon ECS-Container.

`value`

Typ: Zeichenfolge

Erforderlich: Ja

Der Amazon-Ressourcenname (ARN) des Amazon S3-Objekts, das die Umgebungsvariablendatei enthält.

`type`

Typ: Zeichenfolge

Erforderlich: Ja

Dateityp, der verwendet werden muss. Der einzige unterstützte Wert ist `s3`.

`environment`

Typ: Objekt-Array

Erforderlich: Nein

Die Umgebungsvariablen, die an einen Container übergeben werden. Dieser Parameter ordnet zu `Env` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--env` für die docker run zu.

> **Important**
>
> Die Verwendung von Klartext-Umgebungsvariablen für sensitive Informationen (wie etwa Zugangsdaten) wird nicht empfohlen.

`name`

Typ: Zeichenfolge

Erforderlich: ja, wenn `environment` verwendet wird

Der Name der Umgebungsvariable.

`value`

Typ: Zeichenfolge

Erforderlich: ja, wenn `environment` verwendet wird

Der Wert der Umgebungsvariable.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Typ: Objekt-Array

Erforderlich: Nein

Ein Objekt, durch das das Secret dargestellt wird, das dem Container zur Verfügung gestellt werden soll. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

`name`

Typ: Zeichenfolge

Erforderlich: Ja

Der Wert, der als Umgebungsvariable auf dem Container eingestellt werden soll.

`valueFrom`

Typ: Zeichenfolge

Erforderlich: Ja

Das Geheimnis, das dem Container exponiert werden muss. Die unterstützten Werte sind entweder der vollständige Amazon Resource Name (ARN) des AWS Secrets Manager Secrets oder der vollständige ARN des Parameters im AWS Systems Manager Parameter Store.

> **Note**
>
> Wenn der Systems Manager Parameter Store-Parameter oder der Secrets Manager Manager-Parameter in derselben AWS-Region Datei wie die Aufgabe, die Sie starten, vorhanden ist, können Sie entweder den vollständigen ARN oder den Namen des Secrets verwenden. Wenn der Parameter in einer anderen Region existiert, muss der volle ARN angegeben werden.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Network settings (Netzwerkeinstellungen)

`disableNetworking`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Parameter den Wert „true“ aufweist, ist die Netzwerkfunktion innerhalb des Containers deaktiviert. Dieser Parameter ist im Bereich Create a container der Docker Remote API der Option `NetworkDisabled` zugeordnet.

> **Note**
>
> Dieser Parameter wird nicht für Windows-Container oder Aufgaben unterstützt, die den `awsvpc`-Netzwerkmodus verwenden.

Der Standardwert ist `false`.

```
"disableNetworking": true|false
```

`links`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Über den Parameter `link` können Container miteinander kommunizieren, ohne dass Port-Zuweisungen nötig sind. Dieser Parameter wird nur unterstützt, wenn der Netzwerkmodus einer Aufgabendefinition auf `bridge` gesetzt ist. Das Konstrukt `name:internalName` ist analog zu `name:alias` in Docker-Verbindungen. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig. Weitere Informationen zur Verbindung von Docker-Containern finden Sie unter https://docs.docker.com/engine/userguide/networking/default_network/dockerlinks/. Dieser Parameter ordnet zu `Links` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--link` für die docker run zu.

> **Note**
>
> Dieser Parameter wird nicht für Windows-Container oder Aufgaben unterstützt, die den `awsvpc`-Netzwerkmodus verwenden.

> **Important**
>
> Container, die sich auf derselben Container-Instance befinden, können miteinander kommunizieren, ohne dass Verbindungen oder Host-Port-Zuweisungen erforderlich sind. Die Netzwerkisolierung auf einer Container-Instance wird durch Sicherheitsgruppen und VPC-Einstellungen gesteuert.

```
"links": ["name:internalName", ...]
```

`hostname`

Typ: Zeichenfolge

Erforderlich: Nein

Der Hostname, der für Ihren Container verwendet werden soll. Dieser Parameter ordnet zu `Hostname` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--hostname` für die docker run zu.

> **Note**
>
> Wenn Sie den Netzwerkmodus `awsvpc` verwenden, wird der Parameter `hostname` nicht unterstützt.

```
"hostname": "string"
```

`dnsServers`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Eine List der DNS-Server, die dem Container bereitgestellt werden. Dieser Parameter ordnet zu `Dns` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--dns` für die docker run zu.

> **Note**
>
> Dieser Parameter wird nicht für Windows-Container oder Aufgaben unterstützt, die den `awsvpc`-Netzwerkmodus verwenden.

```
"dnsServers": ["string", ...]
```

`dnsSearchDomains`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Muster: ^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$

Eine List der DNS-Such-Domains, die dem Container bereitgestellt werden. Dieser Parameter ordnet zu `DnsSearch` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--dns-search` für die docker run zu.

> **Note**
>
> Dieser Parameter wird für Windows-Container oder -Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, nicht unterstützt.

```
"dnsSearchDomains": ["string", ...]
```

`extraHosts`

Typ: Objekt-Array

Erforderlich: Nein

Eine Liste der Hostnamen und IP-Adresszuordnungen, die an die Datei `/etc/hosts` auf dem Container angefügt werden.

Dieser Parameter ordnet zu `ExtraHosts` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--add-host` für die docker run zu.

> **Note**
>
> Dieser Parameter wird für Windows-Container oder -Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, nicht unterstützt.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `extraHosts` verwendet werden

Der Hostname, der im Eintrag `/etc/hosts` verwendet werden soll.

`ipAddress`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `extraHosts` verwendet werden

Die IP-Adresse, die im Eintrag `/etc/hosts` verwendet werden soll.

Speicher und Protokollierung

`readonlyRootFilesystem`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Parameter den Wert „true“ aufweist, erhält der Container Lesezugriff auf das Root-Dateisystem. Dieser Parameter ordnet zu `ReadonlyRootfs` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--read-only` für die docker run zu.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

Der Standardwert ist `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Typ: Objekt-Array

Erforderlich: Nein

Die Bereitstellungspunkte für die Datenvolumes in Ihrem Container. Dieser Parameter ordnet zu `Volumes` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volume` für die docker run zu.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten. Windows-Container können keine Verzeichnisse auf einem anderen Laufwerk mounten, und Bereitstellungspunkte können nicht laufwerksübergreifend verwendet werden. Sie müssen Bereitstellungspunkte angeben, um ein Amazon EBS-Volume direkt an eine Amazon ECS-Aufgabe anzuhängen.

`sourceVolume`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Name des zu mountenden Volumes.

`containerPath`

Typ: Zeichenfolge

Erforderlich: Ja, wenn `mountPoints` verwendet werden

Der Pfad im Container, in dem das Volume bereitgestellt werden soll.

`readOnly`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume. Der Standardwert ist `false`.

`volumesFrom`

Typ: Objekt-Array

Erforderlich: Nein

Die Daten-Volumens, die von einem anderen Container gemountet werden sollen. Dieser Parameter ordnet zu `VolumesFrom` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--volumes-from` für die docker run zu.

`sourceContainer`

Typ: Zeichenfolge

Erforderlich: ja, wenn `volumesFrom` verwendet wird

Der Name des Containers, von dem die Volumes gemountet werden.

`readOnly`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Wert `true` lautet, verfügt der Container über schreibgeschützten Zugriff auf das Volume. Lautet der Wert `false`, dann verfügt der Container über Schreibzugriff auf das Volume. Der Standardwert ist `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Typ: LogConfigurationObjekt

Erforderlich: Nein

Die Angabe der Protokollkonfiguration für den Container.

Beispiele für Aufgabendefinitionen, die eine Protokollkonfiguration verwenden, finden Sie unter Beispiel für Amazon ECS-Aufgabendefinitionen.

Dieser Parameter ordnet zu `LogConfig` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--log-driver` für die `docker run` zu. Standardmäßig verwenden Container den gleichen Protokolltreiber wie der Docker-Daemon. Allerdings kann der Container einen anderen Protokolltreiber als der Docker-Daemon verwenden, indem Sie einen Protokolltreiber mit diesem Parameter in der Container-Definition angeben. Um für einen Container einen anderen Protokolltreiber zu verwenden, muss das Protokollsystem auf der Container-Instance (oder einem anderen Protokollserver für die Remote-Protokollierung) ordnungsgemäß konfiguriert sein. Weitere Informationen zu den Optionen für die verschiedenen unterstützten Protokolltreiber finden Sie unter Protokolltreiber konfigurieren in der Docker-Dokumentation.

Beachten Sie die folgenden Punkte, wenn eine Protokollkonfiguration für Ihre Container angegeben ist:

- Amazon ECS unterstützt einen Teil der Protokolltreiber, die für den Docker-Daemon verfügbar sind. Es ist möglich, dass in zukünftigen Releases des Amazon-ECS-Container-Agenten weitere Protokolltreiber zur Verfügung stehen.
- Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden.
- Bei Aufgaben mit dem EC2-Starttyp muss der auf einer Container-Instance ausgeführte Amazon-ECS-Container-Agent die auf dieser Instance verfügbaren Protokolltreiber mit der Umgebungsvariablen `ECS_AVAILABLE_LOGGING_DRIVERS` registrieren, bevor in der Instance platzierte Container diese Protokollkonfigurationsoptionen verwenden können. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.
- Für Aufgaben, die den Starttyp Fargate verwenden, müssen Sie zusätzliche Software außerhalb der Aufgabe installieren. Beispiel: Die Fluentd-Ausgangsaggregatoren oder ein Remote-Host, auf dem Logstash ausgeführt wird, um Gelf-Protokolle dorthin zu senden.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","logentries","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Typ: Zeichenfolge

Zulässige Werte: `"awslogs","fluentd","gelf","json-file","journald","logentries","splunk","syslog","awsfirelens"`

Erforderlich: ja, wenn `logConfiguration` verwendet wird

Der für den Container zu verwendende Protokolltreiber. Standardmäßig sind die zuvor aufgeführten gültigen Werte Protokolltreiber, mit denen der Amazon-ECS-Container-Agent kommunizieren kann.

Für Aufgaben mit dem Fargate-Starttyp lauten die unterstützten Protokolltreiber `awslogs`, `splunk` und `awsfirelens`.

Für Aufgaben mit dem EC2-Starttyp lauten die unterstützten Protokolltreiber `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries`, `syslog`, `splunk` und `awsfirelens`.

Weitere Informationen zur Verwendung des `awslogs` Protokolltreibers in Aufgabendefinitionen zum Senden Ihrer Container-Logs an CloudWatch Logs finden Sie unterAmazon ECS-Protokolle senden an CloudWatch .

Weitere Informationen zur Verwendung des Protokolltreibers `awsfirelens` finden Sie unter Routing benutzerdefinierter Protokolle.

> **Note**
>
> Wenn Sie einen benutzerdefinierten Treiber haben, der nicht aufgeführt ist, können Sie das Amazon ECS-Container-Agent-Projekt, das verfügbar ist, forken GitHub und es so anpassen, dass es mit diesem Treiber funktioniert. Wir möchten Sie bitten, uns eventuelle Änderungswünsche mitzuteilen. Allerdings unterstützen wir derzeit nicht die Ausführung modifizierter Kopien dieser Software.

Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden.

`options`

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Die Schlüssel/Wert-Map der Konfigurationsoptionen, die an den Protokolltreiber gesendet werden sollen.

Wenn Sie Protokolle FireLens zur Protokollspeicherung und Analyse an ein AWS Partner Network Ziel AWS-Service oder weiterleiten, können Sie die `log-driver-buffer-limit` Option so einstellen, dass die Anzahl der Ereignisse begrenzt wird, die im Speicher zwischengespeichert werden, bevor sie an den Protokoll-Router-Container gesendet werden. Es kann helfen, ein potenzielles Problem mit dem Verlust von Protokollen zu beheben,

da ein hoher Durchsatz dazu führen könnte, dass der Speicher für Puffer innerhalb von Docker ausgeht. Weitere Informationen finden Sie unter the section called “Konfiguration von Protokollen für hohen Durchsatz”.

Für diesen Parameter muss Ihre Docker Remote API Version 1.19 oder höher in Ihrer Container-Instance verwenden.

`secretOptions`

Typ: Objekt-Array

Erforderlich: Nein

Ein Objekt, welches das Secret darstellt, der an die Protokollkonfiguration übergeben werden soll. Zu den Secrets, die in der Protokollkonfiguration verwendet werden, können ein Authentifizierungs-Token, ein Zertifikat oder ein Verschlüsselungsschlüssel gehören. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

`name`

Typ: Zeichenfolge

Erforderlich: Ja

Der Wert, der als Umgebungsvariable auf dem Container eingestellt werden soll.

`valueFrom`

Typ: Zeichenfolge

Erforderlich: Ja

Das Geheimnis zur Bereitstellung der Protokollkonfiguration des Containers.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
```

```
   "valueFrom": "/ecs/logconfig/splunkcred"
  }]
 }
```

`firelensConfiguration`

Typ: Objekt FirelensConfiguration

Erforderlich: Nein

Die FireLens Konfiguration für den Container. Dies wird verwendet, um einen Protokollrouter für Container-Protokolle anzugeben und zu konfigurieren. Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Die Schlüssel/Wert-Map der Optionen, die bei der Konfiguration des Protokoll-Routers verwendet werden sollen. Dieses Feld ist optional und kann verwendet werden, um eine benutzerdefinierte Konfigurationsdatei anzugeben oder dem Protokollereignis zusätzliche Metadaten hinzuzufügen, z. B. Aufgaben-, Aufgabendefinitions-, Cluster- und Container-Instance-Details. Wenn angegeben, ist die zu verwendende Syntax `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::mybucket/fluent.conf|filepath"}`. Weitere Informationen finden Sie unter Beispiel für eine Amazon ECS-Aufgabendefinition: Protokolle weiterleiten an FireLens.

`type`

Typ: Zeichenfolge

Erforderlich: Ja

Der zu verwendende Protokollrouter. Die gültigen Werte sind `fluentd` und `fluentbit`.

Sicherheit

Weitere Informationen zur Containersicherheit finden Sie im Abschnitt Aufgaben- und Containersicherheit im Leitfaden für bewährte Methoden für Amazon-ECS.

`credentialSpecs`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Eine Liste von ARNs in SSM oder Amazon S3 für eine Datei für Anmeldeinformationsspezifikationen (`CredSpec`), die den Container für die Active-Directory-Authentifizierung konfiguriert. Es wird empfohlen, diesen Parameter anstelle von `dockerSecurityOptions` zu verwenden. Die maximale Anzahl von ARNs ist 1.

Es gibt zwei Formate für jeden ARN.

credentialspecdomainless:MyARN

Sie verwenden `credentialspecdomainless:MyARN`, um `CredSpec` einen zusätzlichen Abschnitt für ein Secret in Secrets Manager bereitzustellen. Sie geben die Anmeldeinformationen für die Domain im Secret an.

Jede Aufgabe, die auf einer beliebigen Container-Instance ausgeführt wird, kann verschiedenen Domains beitreten.

Sie können dieses Format verwenden, ohne die Container-Instance mit einer Domain zu verbinden.

credentialspec:MyARN

Sie verwenden `credentialspec:MyARN`, um `CredSpec` für eine einzelne Domain bereitzustellen.

Sie müssen die Container-Instance mit der Domain verbinden, bevor Sie Aufgaben starten, die diese Aufgabendefinition verwenden.

Ersetzen Sie in beiden Formaten `MyARN` durch den ARN in SSM oder Amazon S3.

Die `credspec` muss im Secrets Manager einen ARN für ein Secret angeben, das den Benutzernamen, das Passwort und die Domain enthält, zu der eine Verbindung hergestellt werden soll. Aus Sicherheitsgründen ist die Instance bei der domainlosen Authentifizierung nicht mit der Domain verbunden. Andere Anwendungen auf der Instance können die domainlosen Anmeldeinformationen nicht verwenden. Sie können diesen Parameter verwenden, um Aufgaben auf derselben Instance auszuführen, auch wenn die Aufgaben mit verschiedenen Domains verbunden werden müssen. Weitere Informationen finden Sie unter Verwenden von gMSAs für Windows-Container und Verwenden von gMSAs für Linux-Container.

`privileged`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Parameter den Wert „true“ aufweist, erhält der Container erhöhte Berechtigungen auf der Host-Container-Instance (ähnlich wie der `root`-Benutzer). Wir raten davon ab, Container mit `privileged` auszuführen. In den meisten Fällen können Sie die genauen Berechtigungen angeben, die Sie benötigen, indem Sie die spezifischen Parameter verwenden, anstatt `privileged` zu verwenden.

Dieser Parameter ordnet zu `Privileged` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--privileged` für die docker run zu.

> **Note**
>
> Dieser Parameter wird für Windows-Container oder -Aufgaben mit dem Starttyp Fargate nicht unterstützt.

Der Standardwert ist `false`.

```
"privileged": true|false
```

`user`

Typ: Zeichenfolge

Erforderlich: Nein

Der Benutzer, der im Container verwendet werden soll. Dieser Parameter ordnet zu `User` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--user` für die docker run zu.

> **Important**
>
> Wenn Sie Aufgaben im Netzwerkmodus `host` ausführen, sollten Sie Container nicht über den Root-Benutzer (UID 0) ausführen. Verwenden Sie als bewährte Sicherheitsmethode immer einen Nicht-Root-Benutzer.

Sie können den `user` mit den folgenden Formaten angeben. Wenn Sie eine User Identifier (UID, Benutzerbezeichner) und Group Identifier (GID, Gruppenbezeichner) angeben, müssen Sie sie als positive Ganzzahl angeben.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

```
"user": "string"
```

`dockerSecurityOptions`

Typ: Zeichenfolgen-Array

Gültige Werte: „no-new-privileges“ | „AppArmor:Profile“ | „label: *value*" | „credentials spec:“ *CredentialSpecFilePath*

Erforderlich: Nein

Eine Liste von Zeichenfolgen für die Bereitstellung benutzerdefinierter Konfigurationen für mehrere Sicherheitssysteme. Weitere Informationen zu gültigen Werten finden Sie unter Sicherheitskonfiguration für die Docker-Ausführung. Dieses Feld ist für Container in Aufgaben mit dem Starttyp Fargate nicht gültig.

Für Linux-Aufgaben auf EC2 kann dieser Parameter verwendet werden, um auf benutzerdefinierte Labels für mehrstufige Sicherheitssysteme von SELinux und AppArmor zu verweisen.

Für alle Aufgaben auf EC2 kann dieser Parameter verwendet werden, um auf eine Anmeldeinformationsspezifikationsdatei zu verweisen, die einen Container für die Active Directory-Authentifizierung konfiguriert. Weitere Informationen finden Sie unter Erfahren Sie, wie Sie GMSAs für EC2-Windows-Container für Amazon ECS verwenden und Verwendung gMSA für Linux EC2-Container auf Amazon ECS.

Dieser Parameter ordnet zu `SecurityOpt` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--security-opt` für die docker zu.

```
"dockerSecurityOptions": ["string", ...]
```

> **Note**
>
> Der auf einer Container-Instance ausgeführte Amazon-ECS-Container-Agent muss mit der Umgebungsvariablen `ECS_SELINUX_CAPABLE=true` oder `ECS_APPARMOR_CAPABLE=true` registriert werden, bevor in der Instance platzierte Container diese Sicherheitsoptionen verwenden können. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Ressourcenlimits

`ulimits`

Typ: Objekt-Array

Erforderlich: Nein

Eine Liste von `ulimit`-Werten, die für einen Container definiert werden sollen. Dieser Wert überschreibt die Standardeinstellung für das Ressourcenkontingent für das Betriebssystem.

Dieser Parameter ordnet zu `Ulimits` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--ulimit` für die docker run zu.

Amazon-ECS-Aufgaben, die auf Fargate gehosted werden, verwenden die Standardwerte für Ressourcenlimits, die vom Betriebssystem gesetzt wurden. Ausgenommen ist der Ressourcenlimitparameter `nofile`. Das Ressourcenlimit `nofile` beschränkt die Anzahl der geöffneten Dateien, die ein Container verwenden kann. Auf Fargate ist die `nofile`-Standardeinstellung für die weiche Grenze `1024` und die harte Grenze `65535`. Sie können die Werte beider Grenzwerte auf bis zu `1048576` festlegen. Weitere Informationen finden Sie unter Aufgabenressourcenlimits.

Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Typ: Zeichenfolge

Zulässige Werte: `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Erforderlich: Ja, wenn `ulimits` verwendet werden

Der `type` des `ulimit`-Werts.

`hardLimit`

Typ: Ganzzahl

Erforderlich: Ja, wenn `ulimits` verwendet werden

Das harte Limit für den `ulimit`-Typ.

`softLimit`

Typ: Ganzzahl

Erforderlich: Ja, wenn `ulimits` verwendet werden

Das weiche Limit für den `ulimit`-Typ.

Docker-Bezeichnungen

`dockerLabels`

Typ: Abbildung einer Zeichenfolge auf eine Zeichenfolge

Erforderlich: Nein

Eine Schlüssel-/Wertzuordnung von Bezeichnungen, die dem Container hinzugefügt werden sollen. Dieser Parameter ordnet zu `Labels` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--label` für die docker run zu.

Für diesen Parameter muss Ihre Docker Remote API Version 1.18 oder höher in Ihrer Container-Instance verwenden.

```
"dockerLabels": {"string": "string"
      ...}
```

Andere Parameter für die Containerdefinition

Die folgenden Parameter für die Containerdefinition können beim Registrieren von Aufgabendefinitionen in der Amazon-ECS-Konsole mit der Option Configure via JSON (Über JSON konfigurieren) verwendet werden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Themen

Linux-Parameter

`linuxParameters`

Typ: LinuxParameters Objekt

Erforderlich: Nein

Linux-spezifische Optionen, die auf den Container angewendet werden, wie z. B. KernelCapabilities

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Typ: KernelCapabilities Objekt

Erforderlich: Nein

Die Linux-Funktionen für den Container, die zur von Docker bereitgestellten Standardkonfiguration hinzugefügt oder daraus entfernt werden. Weitere Informationen zu

den Standardfunktionen und anderen verfügbaren Funktionen finden Sie unter Laufzeit-Berechtigungen und Linux-Funktionen in der Docker-Referenz Docker ausführen. Weitere Informationen zu diesen Linux-Funktionen finden Sie auf der Linux-Handbuchseite Funktionen (7).

`add`

Typ: Zeichenfolgen-Array

Zulässige Werte: `"ALL" | "AUDIT_CONTROL" | "AUDIT_READ" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Erforderlich: Nein

Die Linux-Funktionen für den Container, die zu der von Docker bereitgestellten Standardkonfiguration hinzugefügt werden sollen. Dieser Parameter ist `CapAdd` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--cap-add` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Aufgaben, die auf Fargate gestartet werden, unterstützen lediglich das Hinzufügen der `SYS_PTRACE`-Kernelfunktion.

`add`

Typ: Zeichenfolgen-Array

Zulässige Werte: `"SYS_PTRACE"`

Erforderlich: Nein

Die Linux-Funktionen für den Container, die zu der von Docker bereitgestellten Standardkonfiguration hinzugefügt werden sollen. Dieser Parameter ist `CapAdd` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--cap-add` für die Docker-Ausführung zugeordnet.

`drop`

Typ: Zeichenfolgen-Array

Zulässige Werte: `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Erforderlich: Nein

Die Linux-Funktionen für den Container, die aus der von Docker bereitgestellten Standardkonfiguration entfernt werden sollen. Dieser Parameter ist `CapDrop` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--cap-drop` für die Docker-Ausführung zugeordnet.

`devices`

Alle Host-Geräte, die dem Container zur Verfügung gestellt werden sollen. Dieser Parameter ist `Devices` im Abschnitt Einen Container erstellen der Docker-Remote-API und der Option `--device` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Der `devices`-Parameter wird nicht unterstützt, wenn Sie den Starttyp Fargate oder Windows-Container verwenden.

Typ: Array von Device-Objekten

Erforderlich: Nein

`hostPath`

Der Gerätepfad auf der Hostcontainer-Instance.

Typ: Zeichenfolge

Erforderlich: Ja

`containerPath`

Der Pfad im Container, der dem Hostgerät verfügbar gemacht werden soll.

Typ: Zeichenfolge

Erforderlich: Nein

`permissions`

Die expliziten Berechtigungen, die dem Container für das Gerät bereitgestellt werden sollen. Standardmäßig hat der Container Berechtigungen für `read`, `write`und `mknod` auf dem Gerät.

Typ: Zeichenfolgen-Array

Zulässige Werte: `read` | `write` | `mknod`

`initProcessEnabled`

Führen Sie einen `init`-Prozess innerhalb des Containers aus, der Signalen weiterleitet und Prozesse aufnimmt. Dieser Parameter wird der Option `--init` für die Docker-Ausführung zugeordnet.

Für diesen Parameter muss Ihre Docker Remote API Version 1.25 oder höher in Ihrer Container-Instance verwenden.

`maxSwap`

Die Gesamtmenge des Auslagerungsspeichers (in MiB), den ein Container verwenden kann. Dieser Parameter ist in die Option `--memory-swap` zur Docker-Ausführung übersetzt, wobei der Wert die Summe aus dem Container-Speicher und dem Wert `maxSwap` ist.

Wenn als `maxSwap`-Wert `0` angegeben wird, verwendet der Container keine Auslagerung. Zulässige Werte sind `0` oder eine beliebige positive Ganzzahl. Wenn der Parameter `maxSwap` weggelassen wird, verwendet der Container die Swap-Konfiguration für die Container-

Instance, auf der er ausgeführt wird. Es muss ein Wert für `maxSwap` festgelegt werden, damit der Parameter `swappiness` verwendet werden kann.

> **Note**
>
> Wenn Sie Aufgaben verwenden, die den Starttyp Fargate nutzen, wird der Parameter `maxSwap` nicht unterstützt.

`sharedMemorySize`

Der Wert für die Größe des `/dev/shm`-Volumes (in MiB). Dieser Parameter wird der Option `--shm-size` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Wenn Sie Aufgaben verwenden, die den Starttyp Fargate nutzen, wird der Parameter `sharedMemorySize` nicht unterstützt.

Typ: Ganzzahl

`swappiness`

Auf diese Weise können Sie diesen Parameter verwenden, um Speicherauslagerungsverhalten eines Containers zu optimieren. Ein `swappiness`-Wert von `0` führt dazu, dass kein Auslagern erfolgt, sofern nicht erforderlich. Ein `swappiness`-Wert von `100` führt dazu, dass Seiten häufig ausgelagert werden. Akzeptierte Werte sind Ganzzahlen zwischen `0` und `100`. Wenn Sie keinen Wert angeben, wird der Standardwert `60` verwendet. Darüber hinaus wird dieser Parameter ignoriert, wenn Sie keinen Wert für `maxSwap` angeben. Dieser Parameter wird der Option `--memory-swappiness` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Wenn Sie Aufgaben verwenden, die den Starttyp Fargate nutzen, wird der Parameter `swappiness` nicht unterstützt.
> Wenn Sie Aufgaben auf Amazon Linux 2023 verwenden, wird der `swappiness`-Parameter nicht unterstützt.

`tmpfs`

Der Container-Pfad, Mount-Optionen und Maximalgröße (in MB) des tmpfs-Mounts. Dieser Parameter wird der Option `--tmpfs` für die Docker-Ausführung zugeordnet.

> **Note**
>
> Wenn Sie Aufgaben verwenden, die den Starttyp Fargate nutzen, wird der Parameter `tmpfs` nicht unterstützt.

Typ: Array von Tmpfs-Objekten

Erforderlich: Nein

`containerPath`

Der absolute Dateipfad, unter dem die tmpfs-Volumes gemountet werden.

Typ: Zeichenfolge

Erforderlich: Ja

`mountOptions`

Die Liste der Mount-Optionen für das tmpfs-Volume.

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Zulässige Werte: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

Die maximale Größe (in MiB) des tmpfs-Volumes.

Typ: Ganzzahl

Erforderlich: Ja

Container-Abhängigkeit

`dependsOn`

Typ: Array von ContainerDependency-Objekten

Erforderlich: Nein

Die für das Startup und Herunterfahren des Containers definierten Abhängigkeiten. Ein Container kann mehrere Abhängigkeiten enthalten. Wenn eine Abhängigkeit für das Startup und das Herunterfahren des Containers definiert ist, ist sie reserviert. Ein Beispiel finden Sie unter Container-Abhängigkeit.

> **Note**
>
> Wenn ein Container keine Abhängigkeitsbeschränkung erfüllt oder ein Timeout vor dem Erfüllen der Einschränkung erfüllt, führt Amazon ECS abhängige Container nicht in den nächsten Zustand.

Für Amazon-ECS-Aufgaben, die auf Amazon-EC2-Instances gehostet werden, benötigen die Instances mindestens Version `1.26.0` des Container-Agents, um Container-Abhängigkeiten zu aktivieren. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten. Wenn Sie das Amazon-ECS-optimierte Amazon Linux-AMI verwenden, benötigt Ihre Instance mindestens Version `1.26.0-1` des `ecs-init`-Pakets. Wenn Ihre Container-Instances mit Version `20190301` oder höher gestartet werden, enthalten sie die erforderlichen Versionen des Container-Agenten und von `ecs-init`. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Für Amazon-ECS-Aufgaben, die auf Fargate gehostet werden, erfordert dieser Parameter, dass die Aufgabe oder der Service die Plattformversion `1.3.0` oder später (Linux) oder `1.0.0` (Windows) verwendet.

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Typ: Zeichenfolge

Erforderlich: Ja

Der Container-Name, der die angegebene Bedingung erfüllen muss.

`condition`

Typ: Zeichenfolge

Erforderlich: Ja

Die Abhängigkeitsbedingung des Containers. Der folgenden Tabelle sind die verfügbaren Bedingungen und deren Verhalten zu entnehmen:

- `START` – Diese Bedingung emuliert das heutige Verhalten von Links und Volumes. Die Bedingung überprüft, ob ein abhängiger Container gestartet wurde, bevor das Starten anderer Container zugelassen wird.
- `COMPLETE` – Diese Bedingung überprüft, ob ein abhängiger Container vollständig ausgeführt (beendet) wurde, bevor das Starten anderer Container zugelassen wird. Dies kann für nicht benötigte Container hilfreich sein, die ein Skript ausführen und dann beendet werden. Diese Bedingung kann nicht für einen essentiellen Container festgelegt werden.
- `SUCCESS` – Diese Bedingung ist mit `COMPLETE` identisch, erfordert aber auch, dass der Container mit dem Status `zero` beendet wird. Diese Bedingung kann nicht für einen essentiellen Container festgelegt werden.
- `HEALTHY` – Diese Bedingung überprüft, ob der abhängige Container seine Zustandsprüfung bestanden hat, bevor das Starten anderer Container zugelassen wird. Dies setzt voraus, das für den abhängigen Container Zustandsprüfungen in der Aufgabendefinition konfiguriert wurden. Diese Bedingung wird nur beim Startup der Aufgabe bestätigt.

Container-Timeouts

`startTimeout`

Typ: Ganzzahl

Erforderlich: Nein

Beispielwerte: `120`

Zeitdauer (in Sekunden), die gewartet wird, bevor die Auflösung von Abhängigkeiten für einen Container aufgegeben wird.

Angenommen, Sie geben zwei Container in einer Aufgabendefinition an und `containerA` ist abhängig davon, dass `containerB` den Status `COMPLETE`, `SUCCESS` oder `HEALTHY` erreicht. Wenn ein `startTimeout`-Wert für `containerB` angegeben ist und es den gewünschten Status nicht innerhalb dieses Zeitraums erreicht, wird `containerA` nicht gestartet.

> **Note**
>
> Wenn ein Container keine Abhängigkeitsbeschränkung erfüllt oder ein Timeout vor dem Erfüllen der Einschränkung erfüllt, führt Amazon ECS abhängige Container nicht in den nächsten Zustand.

Für Amazon-ECS-Aufgaben, die auf Fargate gehostet werden, erfordert dieser Parameter, dass die Aufgabe oder der Service die Plattformversion `1.3.0` oder höher (Linux) verwendet. Der Höchstwert beträgt 120 Sekunden.

`stopTimeout`

Typ: Ganzzahl

Erforderlich: Nein

Beispielwerte: `120`

Dauer (in Sekunden), die gewartet werden soll, bevor der Container zwangsweise beendet wird, wenn er nicht normal beendet wird.

Für Amazon-ECS-Aufgaben, die auf Fargate gehostet werden, erfordert dieser Parameter, dass die Aufgabe oder der Service die Plattformversion `1.3.0` oder höher (Linux) verwendet.

Wenn der Parameter nicht angegeben wird, wird der Standardwert 30 Sekunden verwendet. Der Höchstwert beträgt 120 Sekunden.

Bei Aufgaben, für die der EC2-Starttyp verwendet wird, wird, wenn der Parameter `stopTimeout` nicht angegeben ist, wird der für die Konfigurationsvariable `ECS_CONTAINER_STOP_TIMEOUT` des Amazon-ECS-Container-Agenten festgelegte Wert verwendet. Wenn weder der Parameter `stopTimeout` noch die Agent-Konfigurationsvariable `ECS_CONTAINER_STOP_TIMEOUT` festgelegt ist, werden die Standardwerte von 30 Sekunden für Linux-Container und 30 Sekunden für Windows-Container verwendet. Container-Instances benötigen zur Aktivierung eines Timeout-Wertes für das Stoppen des Containers mindestens Version 1.26.0 des Container-Agenten. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten. Wenn Sie das Amazon-ECS-optimierte Amazon Linux-AMI verwenden, benötigt Ihre Instance mindestens Version 1.26.0-1 des `ecs-init`-Pakets. Wenn Ihre Container-Instances mit Version `20190301` oder höher gestartet werden, enthalten sie die erforderlichen Versionen des Container-Agenten und von `ecs-init`. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Systemkontrollen

`systemControls`

Typ: SystemControl Objekt

Erforderlich: Nein

Eine Liste von Namespace-Kernel-Parametern, die im Container festgelegt werden sollen. Dieser Parameter ordnet zu `Sysctls` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--sysctl` für die docker run zu. Beispielsweise können Sie die `net.ipv4.tcp_keepalive_time`-Einstellung konfigurieren, um Verbindungen mit einer längeren Lebensdauer aufrechtzuerhalten.

Wir empfehlen nicht, netzwerkbezogene `systemControls`-Parameter für mehrere Container in einer einzelnen Aufgabe festzulegen, die auch entweder den `awsvpc`- oder `host`-Netzwerkmodus verwendet. Dies hat die folgenden Nachteile:

- Bei Aufgaben, die den `awsvpc`-Netzwerkmodus einschließlich Fargate verwenden, gelten `systemControls`, die Sie für einen beliebigen Container einstellen, für alle Container in der Aufgabe. Wenn Sie unterschiedliche `systemControls` für mehrere Container innerhalb einer

einzelnen Aufgabe festlegen, werden die `systemControls` des zuletzt gestarteten Containers für alle Container übernommen.

- Für Aufgaben, die den Netzwerkmodus `host` verwenden, wird der Netzwerk-Namespace `systemControls` nicht unterstützt.

Wenn Sie einen IPC-Ressourcen-Namespace für die Container in der Aufgabe einrichten, gelten folgende Bedingungen für Ihre Systemsteuerungen. Weitere Informationen finden Sie unter IPC-Modus.

- Für Aufgaben, die den IPC-Modus `host` verwenden, wird der IPC-Namespace `systemControls` nicht unterstützt.
- Für Aufgaben, die den IPC-Modus `task` verwenden, gelten die Werte des IPC-Namespace `systemControls` für alle Container innerhalb einer Aufgabe.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

> **Note**
>
> Dieser Parameter wird nur für Aufgaben unterstützt, die auf AWS Fargate gehostet werden, wenn die Aufgaben die Plattformversion `1.4.0` oder höher (Linux) verwenden. Dies wird für Windows-Container auf Fargate nicht unterstützt.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Typ: Zeichenfolge

Erforderlich: Nein

Der Namespace-Kernelparameter, für den ein `value` Wert gesetzt werden soll.

Gültige IPC-Namespace-Werte: `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"` und `Sysctls`, die mit `"fs.mqueue.*"` beginnen

Gültige Netzwerk-Namespace-Werte: `Sysctls` beginnend mit `"net.*"`

Alle diese Werte werden von Fargate unterstützt.

`value`

Typ: Zeichenfolge

Erforderlich: Nein

Der Wert für den Namespace-Kernelparameter, der in angegeben ist. `namespace`

Interactive

`interactive`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Parameter `true` ist, können Sie Container-Anwendungen bereitstellen, für die `stdin` oder `tty` zugeordnet werden muss. Dieser Parameter ordnet zu `OpenStdin` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--interactive` für die docker run zu.

Der Standardwert ist `false`.

Pseudo-Terminal

`pseudoTerminal`

Typ: Boolesch

Erforderlich: Nein

Wenn dieser Parameter `true` lautet, wird ein TTY zugeordnet. Dieser Parameter ordnet zu `Tty` im Bereich Erstellen eines Containers der Docker Remote API und der Option `--tty` für die docker run zu.

Der Standardwert ist `false`.

Der Name des Elastic-Inference-Beschleunigers

> **Note**
>
> Ab 15. April 2023 AWS wird Amazon Elastic Inference (EI) keine Neukunden mehr in Amazon Elastic Inference (EI) einbinden und Bestandskunden dabei helfen, ihre Workloads auf Optionen umzustellen, die ein besseres Preis und eine bessere Leistung bieten. Nach dem 15. April 2023 können Neukunden keine Instances mit Amazon EI-Beschleunigern in Amazon SageMaker, Amazon ECS oder Amazon EC2 starten. Kunden, die Amazon EI in den letzten 30 Tagen mindestens einmal genutzt haben, gelten jedoch als aktuelle Kunden und können den Service weiterhin nutzen.

Die Ressourcenanforderung des Elastic-Inference-Beschleunigers für Ihre Aufgabendefinition. Weitere Informationen finden Sie unter Was ist Amazon Elastic Inference? im Amazon Elastic Inference Developer Guide.

Die folgenden Parameter sind in einer Aufgabendefinition zulässig:

`deviceName`

Typ: Zeichenfolge

Erforderlich: Ja

Der Gerätename des Elastic Inference-Beschleunigers. Der `deviceName` muss auch in einer Containerdefinition referenziert werden, siehe `Elastic Inference accelerator`.

`deviceType`

Typ: Zeichenfolge

Erforderlich: Ja

Der zu verwendende Elastic-Inference-Beschleuniger.

Aufgabenplatzierungbedingungen

Wenn Sie eine Aufgabendefinition registrieren, können Sie Aufgabenplatzierungsbedingungen angeben, die festlegen, wie Amazon-ECS-Aufgaben platziert.

Wenn Sie den Starttyp Fargate verwenden, werden keine Platzierungsbedingungen für die Aufgaben unterstützt. Standardmäßig werden Fargate-Aufgaben über Availability Zones verteilt.

Für Aufgaben, die den Starttyp EC2 verwenden, können Sie Einschränkungen festlegen, um Tasks basierend auf Availability Zone, Instance-Typ oder benutzerdefinierten Attributen zu platzieren. Weitere Informationen finden Sie unter Definieren Sie, welche Container-Instances Amazon ECS für Aufgaben verwendet.

Die folgenden Parameter sind in einer Containerdefinition zulässig:

`expression`

Typ: Zeichenfolge

Erforderlich: Nein

Ein Ausdruck in Cluster-Abfragesprache, der auf die Einschränkung anzuwenden ist. Weitere Informationen finden Sie unter Erstellen Sie Ausdrücke, um Container-Instances für Amazon ECS-Aufgaben zu definieren.

`type`

Typ: Zeichenfolge

Erforderlich: Ja

Der Typ der Einschränkung. Verwenden Sie `memberOf`, um die Auswahl auf eine Gruppe gültiger Kandidaten zu beschränken.

Proxykonfiguration

`proxyConfiguration`

Typ: ProxyConfiguration Objekt

Erforderlich: Nein

Die Konfigurationsdetails für den App Mesh-Proxy.

Für Aufgaben mit dem Starttyp EC2 benötigen die Container-Instances mindestens Version 1.26.0 des Container-Agenten und mindestens Version 1.26.0-1 des `ecs-init`-Pakets zum Aktivieren einer Proxy-Konfiguration. Wenn Ihre Container-Instances über die Amazon-ECS-optimierte AMI-Version `20190301` oder höher gestartet werden, enthalten sie die erforderlichen Versionen des Container-Agenten und von `ecs-init`. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Für Aufgaben mit dem Starttyp Fargate erfordert dieses Feature, dass die Aufgabe oder der Service von Plattformversion 1.3.0 oder höher Gebrauch macht.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Typ: Zeichenfolge

Gültige Werte: `APPMESH`

Erforderlich: Nein

Der Proxy-Typ. Der einzige unterstützte Wert ist `APPMESH`.

`containerName`

Typ: Zeichenfolge

Erforderlich: Ja

Der Name des Containers, der als App Mesh-Proxy dient.

`properties`

Typ: Array von KeyValuePaar-Objekten

Erforderlich: Nein

Der dem Container Network Interface (CNI)-Plug-In bereitzustellende Satz von Netzwerkkonfigurationsparametern, die als Schlüssel-Wert-Paare angegeben werden.

- `IgnoredUID` – (Erforderlich) Die Benutzer-ID (UID) des Proxy-Containers gemäß der Definition durch den `user`-Parameter in einer Containerdefinition. Dadurch wird sichergestellt, dass der Proxy eigenen Datenverkehr ignoriert. Wenn `IgnoredGID` angegeben wird, kann dieses Feld leer sein.
- `IgnoredGID` – (Erforderlich) Die Gruppen-ID (GID) des Proxy-Containers gemäß der Definition durch den `user`-Parameter in einer Containerdefinition. Dadurch wird sichergestellt, dass der Proxy eigenen Datenverkehr ignoriert. Wenn `IgnoredUID` angegeben wird, kann dieses Feld leer sein.
- `AppPorts` – (Erforderlich) Die Liste der Ports, welche die Anwendung verwendet. Der Netzwerkdatenverkehr zu diesen Ports wird an den `ProxyIngressPort` und den `ProxyEgressPort` weitergeleitet.
- `ProxyIngressPort` – (Erforderlich) Gibt den Port an, an den eingehender Datenverkehr zu den `AppPorts` geleitet wird.
- `ProxyEgressPort` – (Erforderlich) Gibt den Port an, an den ausgehender Datenverkehr zu den `AppPorts` geleitet wird.
- `EgressIgnoredPorts` – (Erforderlich) Der ausgehende Datenverkehr zu diesen angegebenen Ports wird ignoriert und nicht an den `ProxyEgressPort` umgeleitet. Es kann sich um eine leere Liste handeln.
- `EgressIgnoredIPs` – (Erforderlich) Der ausgehende Datenverkehr zu diesen angegebenen IP-Adressen wird ignoriert und nicht an den `ProxyEgressPort` umgeleitet. Es kann sich um eine leere Liste handeln.

`name`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Schlüssel-Wert-Paares.

`value`

Typ: Zeichenfolge

Erforderlich: Nein

Der Wert des Schlüssel-Wert-Paares.

Datenträger

Wenn Sie eine Aufgabendefinition registrieren, können Sie optional eine Liste von Volumes angeben, die an den Docker Daemon auf einer Container-Instance übergeben werden sollen, auf die dann andere Container auf derselben Container-Instance zugreifen können.

Im Folgenden werden die Daten-Volume-Typen aufgeführt, die verwendet werden können:

- Amazon EBS-Volumes — Bietet kostengünstigen, langlebigen und leistungsstarken Blockspeicher für datenintensive containerisierte Workloads. Sie können 1 Amazon EBS-Volume pro Amazon ECS-Aufgabe anhängen, wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen oder aktualisieren. Amazon EBS-Volumes werden für Linux-Aufgaben unterstützt, die auf Fargate- oder Amazon EC2 EC2-Instances gehostet werden. Weitere Informationen finden Sie unter Verwenden Sie Amazon EBS-Volumes mit Amazon ECS.
- Amazon EFS Volumes: Bietet einen einfachen, skalierbaren und persistenten Dateispeicher für die Verwendung mit Amazon-ECS-Aufgaben. Mit Amazon EFS ist die Speicherkapazität elastisch. Sie wird automatisch erweitert oder verkleinert, wenn Sie Dateien hinzufügen oder entfernen. Ihre Anwendungen verfügen jederzeit über den jeweils erforderlichen Speicherplatz, den sie brauchen. Amazon EFS Volumes werden für Aufgaben unterstützt, die auf Fargate oder Amazon-EC2-Instances gehostet werden. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.
- FSx for Windows File Server Volumes: Bietet vollständig verwaltete Microsoft Windows-Dateiserver. Diese Dateiserver werden von einem Windows-Dateisystem unterstützt. Wenn Sie FSx for Windows File Server zusammen mit Amazon ECS verwenden, können Sie Ihre Windows-Aufgaben mit persistenter, verteilter, freigegebener und statischer Dateispeicher bereitstellen. Weitere Informationen finden Sie unter Verwenden Sie FSx for Windows File Server Windows-Dateiserver-Volumes mit Amazon ECS.

Windows-Container auf Fargate unterstützen diese Option nicht.

- Docker-Volumes — Ein von Docker verwaltetes Volume, das unter `/var/lib/docker/volumes` der Amazon EC2 EC2-Host-Instance erstellt wird. Mit Docker-Volume-Treibern (auch als Plug-Ins bezeichnet) werden die Volumes in externe Speichersysteme wie Amazon EBS integriert. Sie können den integrierten `local`-Volume-Treiber oder den Volume-Treiber eines Drittanbieters verwenden. Docker-Volumes werden nur unterstützt, wenn Aufgaben auf Amazon EC2 EC2-Instances ausgeführt werden. Windows-Container unterstützen nur die Verwendung des `local` Treibers. Um Docker-Volumes zu verwenden, geben Sie eine `dockerVolumeConfiguration` in Ihrer Aufgabendefinition an. Weitere Informationen finden Sie unter Verwenden von Volumes.
- Bind-Mounts — Eine Datei oder ein Verzeichnis auf dem Host-Computer, das in einen Container gemountet ist. Bind-Mount-Host-Volumes werden unterstützt, wenn Aufgaben auf AWS Fargate- oder Amazon EC2 EC2-Instances ausgeführt werden. Um Bind-Mount-Host-Volumes zu verwenden, geben Sie einen `host` und optionalen `sourcePath`-Wert in Ihrer Aufgabendefinition an. Weitere Informationen finden Sie unter Verwenden von Bind-Mounts.

Weitere Informationen finden Sie unter Speicheroptionen für Amazon ECS-Aufgaben.

Die folgenden Parameter sind in einer Containerdefinition zulässig.

`name`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Volumes. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Zahlen, Bindestriche () und Unterstriche (`-`) sind zulässig. _ Auf diesen Namen wird im `sourceVolume` Parameter des Container-Definitionsobjekts verwiesen. `mountPoints`

`host`

Erforderlich: Nein

Der `host`-Parameter wird verwendet, um den Lebenszyklus des Bind-Mounts an die Amazon-EC2-Host-Instance und nicht an die Aufgabe zu binden und dort zu speichern. Wenn der Parameter `host` leer ist, weist der Docker-Daemon einen Hostpfad für Ihr Daten-Volume zu, es wird aber nicht gewährleistet, dass die Daten beibehalten werden, nachdem die damit verknüpften Container nicht mehr ausgeführt werden.

Windows-Container können ganze Verzeichnisse auf dem gleichen Laufwerk wie `$env:ProgramData` mounten.

> **Note**
>
> Der `sourcePath` Parameter wird nur unterstützt, wenn Aufgaben verwendet werden, die auf Amazon EC2 EC2-Instances gehostet werden.

`sourcePath`

Typ: Zeichenfolge

Erforderlich: Nein

Wenn der Parameter `host` verwendet wird, geben Sie einen `sourcePath` an, um den Pfad auf der Amazon-EC2-Host-Instance zu deklarieren, die dem Container bereitgestellt wird. Wenn dieser Parameter leer ist, weist der Docker-Daemon einen Host-Pfad für Sie zu. Wenn der Parameter `host` den Speicherort `sourcePath` enthält, bleibt das Daten-Volume an der angegebenen Position der Amazon-EC2-Host-Instance erhalten, bis Sie es manuell löschen. Wenn der Wert `sourcePath` auf der Amazon-EC2-Host-Instance nicht vorhanden ist, wird er vom Docker-Daemon erstellt. Wenn der Speicherort nicht vorhanden ist, wird der Inhalt des Quellpfadordners exportiert.

`configuredAtLaunch`

Typ: Boolesch

Erforderlich: Nein

Gibt an, ob ein Volume beim Start konfigurierbar ist. Wenn diese Option auf gesetzt ist`true`, können Sie das Volume konfigurieren, wenn Sie eine eigenständige Aufgabe ausführen oder wenn Sie einen Dienst erstellen oder aktualisieren. Wenn diese Option auf gesetzt ist`true`, können Sie in der Aufgabendefinition keine andere Volume-Konfiguration angeben. Dieser Parameter muss auf gesetzt werden, `true` um ein Amazon EBS-Volume für das Anhängen an eine Aufgabe zu konfigurieren. Wenn Sie `configuredAtLaunch` die Volume-Konfiguration auf die Startphase festlegen `true` und diese auf die Startphase verschieben, können Sie Aufgabendefinitionen erstellen, die nicht auf einen Volume-Typ oder auf bestimmte Volume-Einstellungen beschränkt sind. Auf diese Weise können Sie Ihre Aufgabendefinition in

verschiedenen Ausführungsumgebungen wiederverwenden. Weitere Informationen finden Sie unter Amazon EBS-Volumes.

`dockerVolumeConfiguration`

Typ: DockerVolumeKonfigurationsobjekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von Docker-Volumes angegeben. Docker-Volumes werden nur unterstützt, wenn Aufgaben auf EC2-Instances ausgeführt werden. Windows-Container unterstützen nur die Verwendung des `local` Treibers. Um Bind-Mounts zu verwenden, geben Sie stattdessen einen `host` an.

`scope`

Typ: Zeichenfolge

Zulässige Werte: `task` | `shared`

Erforderlich: Nein

Der Bereich für das Docker-Volume, der den Lebenszyklus bestimmt. Docker-Volumes, die auf eine `task` beschränkt sind, werden automatisch beim Starten der Aufgabe bereitgestellt und beim Stoppen dieser vernichtet. Docker-Volumes, die als `shared` angewendet werden, bleiben erhalten, nachdem die Aufgabe gestoppt wird.

`autoprovision`

Typ: Boolesch

Standardwert: `false`

Erforderlich: Nein

Wenn dieser Wert `true` lautet, wird das Docker-Volume erstellt, wenn es nicht bereits vorhanden ist. Dieses Feld wird nur verwendet, wenn `scope` der`shared`. Wenn der `scope` ist`task`, muss dieser Parameter entweder weggelassen oder auf gesetzt werden`false`.

`driver`

Typ: Zeichenfolge

Erforderlich: Nein

Der zu verwendende Docker-Volume-Treiber. Der Treiberwert muss mit dem von Docker bereitgestellten Treibernamen übereinstimmen, da dieser Name für die Aufgabenplatzierung verwendet wird. Wenn der Treiber mithilfe der Docker-Plug-in-CLI installiert wurde, verwenden Sie ihn, `docker plugin ls` um den Treibernamen von Ihrer Container-Instance abzurufen. Wenn der Treiber mit einer anderen Methode installiert wurde, verwenden Sie die Docker-Plug-in-Erkennung, um den Treibernamen abzurufen. Weitere Informationen finden Sie unter Docker-Plug-In-Erkennung. Dieser Parameter ist `Driver` im Bereich Create a volume der Docker-Remote-API und der `--driver`-Option zum `docker volume create` zugewiesen.

`driverOpts`

Typ: Zeichenfolge

Erforderlich: Nein

Eine Übersicht mit Treiberspezifischen Optionen für den Docker-Treiber, die durchgespielt werden sollen. Dieser Parameter ist `DriverOpts` im Bereich Create a volume der Docker-Remote-API und der `--opt`-Option zum `docker volume create` zugewiesen.

`labels`

Typ: Zeichenfolge

Erforderlich: Nein

Benutzerdefinierte Metadaten, die Ihrem Docker-Volume hinzugefügt werden sollen. Dieser Parameter ist `Labels` im Bereich Create a volume der Docker-Remote-API und der `--label`-Option zum `docker volume create` zugewiesen.

`efsVolumeConfiguration`

Typ: VolumeConfigurationEFS-Objekt

Erforderlich: Nein

Dieser Parameter wird nur bei der Verwendung von Amazon EFS-Volumes angegeben.

`fileSystemId`

Typ: Zeichenfolge

Erforderlich: Ja

Die zu verwendende Amazon EFS-Dateisystem-ID.

`rootDirectory`

Typ: Zeichenfolge

Erforderlich: Nein

Das Verzeichnis im Amazon EFS-Dateisystem, das als Stammverzeichnis im Host bereitgestellt werden soll. Wenn dieser Parameter weggelassen wird, wird der Stamm des Amazon EFS-Volumes verwendet. Die Angabe von `/` hat denselben Effekt wie das Weglassen dieses Parameters.

> **Important**
>
> Wenn in der ein EFS-Zugriffspunkt angegeben ist`authorizationConfig`, muss der Stammverzeichnisparameter entweder weggelassen oder auf gesetzt werden`/`, wodurch der auf dem EFS-Zugriffspunkt festgelegte Pfad erzwungen wird.

`transitEncryption`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Gibt an, ob die Verschlüsselung für Amazon-EFS-Daten während der Übertragung zwischen dem Amazon-ECS-Host und dem Amazon-EFS-Server aktiviert werden soll oder nicht. Wenn die Amazon-EFS-IAM-Autorisierung verwendet wird, muss die Transit-Verschlüsselung aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter Verschlüsseln von Daten während der Übertragung im Benutzerhandbuch für Amazon Elastic File System.

`transitEncryptionPort`

Typ: Ganzzahl

Erforderlich: Nein

Der zu verwendende Port zum Senden verschlüsselter Daten zwischen dem Amazon-ECS-Host und dem Amazon EFS-Server. Wenn Sie keinen Port für die Übertragungsverschlüsselung angeben, verwendet die Aufgabe die Portauswahlstrategie, die der Amazon EFS-Mount-Helper verwendet. Weitere Informationen finden Sie unter EFS-Mount-Helfer im Benutzerhandbuch für Amazon Elastic File System.

`authorizationConfig`

Typ: AuthorizationConfigurationEFS-Objekt

Erforderlich: Nein

Die Autorisierungskonfigurationsdetails für das Amazon EFS-Dateisystem.

`accessPointId`

Typ: Zeichenfolge

Erforderlich: Nein

Die zu verwendende Zugriffspunkt-ID. Wenn ein Zugriffspunkt angegeben ist, `efsVolumeConfiguration` muss der Stammverzeichniswert in der entweder weggelassen oder auf gesetzt werden`/`, wodurch der auf dem EFS-Zugriffspunkt festgelegte Pfad erzwungen wird. Wenn ein Zugriffspunkt verwendet wird, muss in `EFSVolumeConfiguration` die Transitverschlüsselung aktiviert sein. Weitere Informationen finden Sie unter Arbeiten mit Amazon EFS-Zugriffspunkten im Amazon Elastic File System-Benutzerhandbuch.

`iam`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Gibt an, ob die Amazon ECS-Aufgaben-IAM-Rolle verwendet werden soll, die in einer Aufgabendefinition definiert ist, wenn das Amazon EFS-Dateisystem bereitgestellt wird. Wenn diese Option aktiviert ist, muss die Transit-Verschlüsselung in `EFSVolumeConfiguration` aktiviert sein. Wenn dieser Parameter nicht angegeben ist, wird der Standardwert `DISABLED` verwendet. Weitere Informationen finden Sie unter IAM-Rollen für Aufgaben.

`FSxWindowsFileServerVolumeConfiguration`

Typ: F-Objekt SxWindows FileServer VolumeConfiguration

Erforderlich: Ja

Dieser Parameter wird angegeben, wenn Sie ein Dateisystem von Amazon FSx für Windows als Aufgabenspeicher verwenden.

`fileSystemId`

Typ: Zeichenfolge

Erforderlich: Ja

Die zu verwendende FSx for Windows File Server Dateisystem-ID.

`rootDirectory`

Typ: Zeichenfolge

Erforderlich: Ja

Das Verzeichnis im FSx for Windows File Server-Dateisystem, das als Stammverzeichnis im Host aufgespielt werden soll.

`authorizationConfig`

`credentialsParameter`

Typ: Zeichenfolge

Erforderlich: Ja

Die Optionen für Autorisierungsanmeldeinformationen.

Optionen:

- Amazon-Ressourcenname (ARN) eines AWS Secrets ManagerGeheimnisses.
- ARN eines AWS Systems ManagerParameters.

`domain`

Typ: Zeichenfolge

Erforderlich: Ja

Ein vollqualifizierter Domänenname, der von einem AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) -Verzeichnis oder einem selbst gehosteten EC2 Active Directory gehostet wird.

Tags

Wenn Sie eine Aufgabendefinition registrieren, können Sie optional Metadatentags angeben, die auf die Aufgabendefinition angewendet werden. Mithilfe von Tags können Sie Ihre Aufgabendefinition kategorisieren und organisieren. Jedes Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert. Sie können beide definieren. Weitere Informationen finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.

> **Important**
>
> Fügen Sie keine personenbezogenen Daten (Personally Identifiable Information, PII) oder andere vertrauliche Informationen in Tags hinzu. Tags sind für viele AWS Dienste zugänglich, auch für die Abrechnung. Tags sind nicht für private oder vertrauliche Daten gedacht.

Die folgenden Parameter sind in einem Tag-Objekt zulässig.

`key`

Typ: Zeichenfolge

Erforderlich: Nein

Ein Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Schlüssel ist eine allgemeine Bezeichnung, die wie eine Kategorie für spezifischere Tag-Werte fungiert.

`value`

Typ: Zeichenfolge

Erforderlich: Nein

Der optionale Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Wert fungiert als Deskriptor in einer Tag-Kategorie (Schlüssel).

Andere Parameter der Aufgabendefinition

Die folgenden Parameter für die Aufgabendefinition können beim Registrieren von Aufgabendefinitionen in der Amazon-ECS-Konsole mit der Option Configure via JSON (Über JSON konfigurieren) verwendet werden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Themen

Flüchtiger Speicher

`ephemeralStorage`

Typ: EphemeralStorage Objekt

Erforderlich: Nein

Die Menge des flüchtigen Speichers (in GB), der für die Aufgabe zugewiesen werden soll. Dieser Parameter wird verwendet, um die Gesamtmenge des flüchtigen Speichers über den Standardbetrag hinaus für Aufgaben zu erweitern, die auf AWS Fargate gehostet werden. Weitere Informationen finden Sie unter the section called “Bind-Mounts”.

> **Note**
>
> Dieser Parameter wird nur für Aufgaben unterstützt, die auf AWS Fargate gehostet werden, wenn die Aufgaben die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` oder höher (Windows) verwenden.

IPC-Modus

`ipcMode`

Typ: Zeichenfolge

Erforderlich: Nein

Der IPC-Ressourcen-Namespace, der für die Container in der Aufgabe verwendet werden soll. Die gültigen Werte sind `host`, `task` oder `none`. Wenn `host` angegeben ist, dann teilen sich alle Container innerhalb der Aufgaben, die den IPC-Modus `host` auf derselben Container-Instance angegeben haben, die gleichen IPC-Ressourcen mit der Amazon EC2-Host-Instance. Wenn `task` angegeben ist, teilen sich alle Container innerhalb der angegebenen Aufgabe die gleichen IPC-Ressourcen. Wenn `none` angegeben ist, dann sind die IPC-Ressourcen innerhalb der Container einer Aufgabe privat und werden nicht mit anderen Containern einer Aufgabe oder der Container-Instance geteilt. Wenn kein Wert angegeben ist, hängt die gemeinsame Nutzung des IPC-Ressourcen-Namespace von der Einstellung des Docker-Daemons in der Container-Instance ab. Weitere Informationen finden Sie unter IPC-Einstellungen in der Docker Run Referenz.

Wenn der IPC-Modus `host` verwendet wird, ist das Risiko einer unerwünschten Exposition des IPC-Namespace erhöht. Weitere Informationen finden Sie unter Docker-Sicherheit.

Wenn Sie im Namespace bezogene Kernelparameter mit `systemControls` für die Container in der Aufgabe festlegen, gilt Folgendes für Ihren IPC-Ressourcen-Namespace. Weitere Informationen finden Sie unter Systemkontrollen.

- Für Aufgaben, die den IPC-Modus `host` verwenden, werden IPC-Namespace bezogene `systemControls` nicht unterstützt.
- Für Aufgaben, die den IPC-Modus `task` verwenden, gelten IPC-Namespace bezogene `systemControls` für alle Container innerhalb einer Aufgabe.

> **Note**
>
> Dieser Parameter wird für Windows-Container oder -Aufgaben mit dem Starttyp Fargate nicht unterstützt.

PID-Modus

`pidMode`

Typ: Zeichenfolge

Zulässige Werte: `host` | `task`

Erforderlich: Nein

Der Prozess-Namespace, der für die Container in der Aufgabe verwendet werden soll. Die gültigen Werte sind `host` und `task`. Auf Fargate für Linux-Container ist `task` der einzige gültige Wert. Für die Überwachung von Beiwagen benötigt `pidMode` möglicherweise Informationen über andere Container, die in der gleichen Aufgabe laufen.

Wenn `host` angegeben ist, teilen sich alle Container innerhalb der Aufgaben, die den PID-Modus `host` auf derselben Container-Instance angegeben haben, denselben Prozess-Namespace mit der Amazon EC2-Host-Instance.

Wenn `task` angegeben ist, teilen sich alle Container innerhalb der angegebenen Aufgabe den gleichen Prozess-Namespace.

Wenn kein Wert angegeben wird, ist der Standard ein privater Namespace für jeden Container. Weitere Informationen finden Sie unter PID-Einstellungen in der Docker Run Referenz.

Wenn der PID-Modus `host` verwendet wird, ist das Risiko einer unerwünschten Exposition des Prozess-Namespace erhöht. Weitere Informationen finden Sie unter Docker-Sicherheit.

> **Note**
>
> Dieser Parameter wird für Windows-Container nicht unterstützt.

> **Note**
>
> Dieser Parameter wird nur für Aufgaben unterstützt, die auf AWS Fargate gehostet werden, wenn die Aufgaben die Plattformversion `1.4.0` oder höher (Linux) verwenden. Dies wird für Windows-Container auf Fargate nicht unterstützt.

Amazon ECS-Aufgabendefinitionsvorlage

Nachfolgend wird eine leere Aufgabendefinitionsvorlage gezeigt. Sie können diese Vorlage verwenden, um Ihre Aufgabendefinition zu erstellen, die dann in den JSON-Eingabebereich der Konsole eingefügt oder in einer Datei gespeichert und mit der AWS CLI `--cli-input-json` Option verwendet werden kann. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionsparameter.

Vorlage für den Amazon EC2 EC2-Starttyp

```
{
    "family": "",
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "none",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {
                "credentialsParameter": ""
            },
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [
                ""
            ],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [
                ""
            ],
            "command": [
                ""
            ],
            "environment": [
                {
                    "name": "",
                    "value": ""
                }
            ],
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
```

```
            }
        ],
        "mountPoints": [
            {
                "sourceVolume": "",
                "containerPath": "",
                "readOnly": true
            }
        ],
        "volumesFrom": [
            {
                "sourceContainer": "",
                "readOnly": true
            }
        ],
        "linuxParameters": {
            "capabilities": {
                "add": [
                    ""
                ],
                "drop": [
                    ""
                ]
            },
            "devices": [
                {
                    "hostPath": "",
                    "containerPath": "",
                    "permissions": [
                        "read"
                    ]
                }
            ],
            "initProcessEnabled": true,
            "sharedMemorySize": 0,
            "tmpfs": [
                {
                    "containerPath": "",
                    "size": 0,
                    "mountOptions": [
                        ""
                    ]
                }
            ],
```

```
                "maxSwap": 0,
                "swappiness": 0
            },
            "secrets": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ],
            "dependsOn": [
                {
                    "containerName": "",
                    "condition": "COMPLETE"
                }
            ],
            "startTimeout": 0,
            "stopTimeout": 0,
            "hostname": "",
            "user": "",
            "workingDirectory": "",
            "disableNetworking": true,
            "privileged": true,
            "readonlyRootFilesystem": true,
            "dnsServers": [
                ""
            ],
            "dnsSearchDomains": [
                ""
            ],
            "extraHosts": [
                {
                    "hostname": "",
                    "ipAddress": ""
                }
            ],
            "dockerSecurityOptions": [
                ""
            ],
            "interactive": true,
            "pseudoTerminal": true,
            "dockerLabels": {
                "KeyName": ""
            },
            "ulimits": [
```

```
                {
                    "name": "nofile",
                    "softLimit": 0,
                    "hardLimit": 0
                }
            ],
            "logConfiguration": {
                "logDriver": "splunk",
                "options": {
                    "KeyName": ""
                },
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "healthCheck": {
                "command": [
                    ""
                ],
                "interval": 0,
                "timeout": 0,
                "retries": 0,
                "startPeriod": 0
            },
            "systemControls": [
                {
                    "namespace": "",
                    "value": ""
                }
            ],
            "resourceRequirements": [
                {
                    "value": "",
                    "type": "InferenceAccelerator"
                }
            ],
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "KeyName": ""
                }
```

```
            }
        }
    ],
    "volumes": [
        {
            "name": "",
            "host": {
                "sourcePath": ""
            },
            "configuredAtLaunch": true,
            "dockerVolumeConfiguration": {
                "scope": "shared",
                "autoprovision": true,
                "driver": "",
                "driverOpts": {
                    "KeyName": ""
                },
                "labels": {
                    "KeyName": ""
                }
            },
            "efsVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "transitEncryption": "DISABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
                }
            },
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "authorizationConfig": {
                    "credentialsParameter": "",
                    "domain": ""
                }
            }
        }
    ],
    "placementConstraints": [
        {
            "type": "memberOf",
```

```
            "expression": ""
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ],
    "cpu": "",
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "pidMode": "task",
    "ipcMode": "task",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ],
    "ephemeralStorage": {
        "sizeInGiB": 0
    },
    "runtimePlatform": {
        "cpuArchitecture": "X86_64",
        "operatingSystemFamily": "WINDOWS_SERVER_20H2_CORE"
    }
}
```

Vorlage für den Fargate-Starttyp

Important

Für den Starttyp Fargate müssen Sie den `operatingSystemFamily` Parameter mit einem der folgenden Werte angeben:

- `LINUX`
- `WINDOWS_SERVER_2019_FULL`
- `WINDOWS_SERVER_2019_CORE`
- `WINDOWS_SERVER_2022_FULL`
- `WINDOWS_SERVER_2022_CORE`

```
{
    "family": "",
    "runtimePlatform": {"operatingSystemFamily": ""},
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "awsvpc",
    "platformFamily": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {"credentialsParameter": ""},
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [""],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [""],
            "command": [""],
            "environment": [
```

```
                {
                    "name": "",
                    "value": ""
                }
            ],
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "",
                    "containerPath": "",
                    "readOnly": true
                }
            ],
            "volumesFrom": [
                {
                    "sourceContainer": "",
                    "readOnly": true
                }
            ],
            "linuxParameters": {
                "capabilities": {
                    "add": [""],
                    "drop": [""]
                },
                "devices": [
                    {
                        "hostPath": "",
                        "containerPath": "",
                        "permissions": ["read"]
                    }
                ],
                "initProcessEnabled": true,
                "sharedMemorySize": 0,
                "tmpfs": [
                    {
                        "containerPath": "",
                        "size": 0,
                        "mountOptions": [""]
                    }
```

```
                ],
                "maxSwap": 0,
                "swappiness": 0
            },
            "secrets": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ],
            "dependsOn": [
                {
                    "containerName": "",
                    "condition": "HEALTHY"
                }
            ],
            "startTimeout": 0,
            "stopTimeout": 0,
            "hostname": "",
            "user": "",
            "workingDirectory": "",
            "disableNetworking": true,
            "privileged": true,
            "readonlyRootFilesystem": true,
            "dnsServers": [""],
            "dnsSearchDomains": [""],
            "extraHosts": [
                {
                    "hostname": "",
                    "ipAddress": ""
                }
            ],
            "dockerSecurityOptions": [""],
            "interactive": true,
            "pseudoTerminal": true,
            "dockerLabels": {"KeyName": ""},
            "ulimits": [
                {
                    "name": "msgqueue",
                    "softLimit": 0,
                    "hardLimit": 0
                }
            ],
            "logConfiguration": {
```

```
                "logDriver": "awslogs",
                "options": {"KeyName": ""},
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "healthCheck": {
                "command": [""],
                "interval": 0,
                "timeout": 0,
                "retries": 0,
                "startPeriod": 0
            },
            "systemControls": [
                {
                    "namespace": "",
                    "value": ""
                }
            ],
            "resourceRequirements": [
                {
                    "value": "",
                    "type": "GPU"
                }
            ],
            "firelensConfiguration": {
                "type": "fluentd",
                "options": {"KeyName": ""}
            }
        }
    ],
    "volumes": [
        {
            "name": "",
            "host": {"sourcePath": ""},
            "configuredAtLaunch":true,
            "dockerVolumeConfiguration": {
                "scope": "task",
                "autoprovision": true,
                "driver": "",
                "driverOpts": {"KeyName": ""},
```

```
                "labels": {"KeyName": ""}
            },
            "efsVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
                }
            }
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "",
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "ephemeralStorage": {"sizeInGiB": 0},
    "pidMode": "task",
    "ipcMode": "none",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ]
}
```

Sie können diese Aufgabendefinitionsvorlage mit dem folgenden AWS CLI -Befehl generieren.

```
aws ecs register-task-definition --generate-cli-skeleton
```

Beispiel für Amazon ECS-Aufgabendefinitionen

Sie können die Beispiele und Ausschnitte kopieren, um mit der Erstellung Ihrer eigenen Aufgabendefinitionen zu beginnen.

Sie können die Beispiele kopieren und dann einfügen, wenn Sie die Option Über JSON konfigurieren in der Konsole verwenden. Stellen Sie sicher, dass Sie die Beispiele anpassen, z. B. durch die Verwendung Ihrer Konto-ID. Sie können die Ausschnitte in Ihre JSON-Aufgabendefinition aufnehmen. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole und Amazon ECS-Aufgabendefinitionsparameter.

Weitere Beispiele für Aufgabendefinitionen finden Sie unter AWS Beispiel-Aufgabendefinitionen unter. GitHub

Themen

- Webserver
- splunkProtokolltreiber
- fluentdProtokolltreiber
- gelfProtokolltreiber
- Workloads auf externen Instanzen
- Amazon ECR-Image- und Aufgabendefinition (IAM-Rolle)
- Einstiegspunkt mit Befehl
- Container-Abhängigkeit
- Beispiele für Windows-Aufgabendefinitionen

Webserver

Nachfolgend finden Sie Beispiele für Aufgabendefinitionen mit Linux-Containern auf dem Fargate-Launchtyp zum Einrichten eines Webservers:

```
{
   "containerDefinitions": [
```

```
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
```

```
}
```

Nachfolgend finden Sie Beispiele für Aufgabendefinitionen mit Windows-Containern auf dem Fargate-Launchtyp zum Einrichten eines Webservers:

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

`splunk`Protokolltreiber

Im folgenden Ausschnitt wird gezeigt, wie Sie den `splunk`-Protokolltreiber in einer Aufgabendefinition verwenden, die die Protokolle an einen Remote-Service sendet. Der Splunk-Token-Parameter wird als geheime Option angegeben, da er als sensible Daten behandelt werden kann. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

```
"containerDefinitions": [{
  "logConfiguration": {
   "logDriver": "splunk",
   "options": {
    "splunk-url": "https://cloud.splunk.com:8080",
    "tag": "tag_name",
   },
   "secretOptions": [{
    "name": "splunk-token",
    "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:splunk-token-
KnrBkD"
}],
```

`fluentd`Protokolltreiber

Im folgenden Ausschnitt wird gezeigt, wie Sie den `fluentd`-Protokolltreiber in einer Aufgabendefinition verwenden, die die Protokolle an einen Remote-Service sendet. Der `fluentd-address`-Wert ist als geheime Option angegeben, da er als sensible Daten behandelt werden kann. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "fluentd",
  "options": {
   "tag": "fluentd demo"
  },
  "secretOptions": [{
   "name": "fluentd-address",
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:fluentd-address-
KnrBkD"
  }]
```

```
 },
 "entryPoint": [],
 "portMappings": [{
             "hostPort": 80,
             "protocol": "tcp",
             "containerPort": 80
             },
             {
  "hostPort": 24224,
  "protocol": "tcp",
  "containerPort": 24224
 }]
}],
```

`gelf`Protokolltreiber

Im folgenden Ausschnitt wird gezeigt, wie Sie den `gelf`-Protokolltreiber in einer Aufgabendefinition verwenden, die Protokolle an einen Remote-Host sendet, auf dem Logstash ausgeführt wird und Gelf-Protokolle als Eingang verwendet werden. Weitere Informationen finden Sie unter `logConfiguration`.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "gelf",
  "options": {
   "gelf-address": "udp://logstash-service-address:5000",
   "tag": "gelf task demo"
  }
 },
 "entryPoint": [],
 "portMappings": [{
   "hostPort": 5000,
   "protocol": "udp",
   "containerPort": 5000
  },
  {
   "hostPort": 5000,
   "protocol": "tcp",
   "containerPort": 5000
  }
 ]
}],
```

Workloads auf externen Instanzen

Verwenden Sie bei der Registrierung einer Amazon-ECS-Aufgabendefinition den `requiresCompatibilities`-Parameter und geben Sie `EXTERNAL` an, der überprüft, ob die Aufgabendefinition kompatibel ist, wenn Amazon-ECS-Workloads auf Ihren externen Instances ausgeführt werden. Wenn Sie die Konsole für die Registrierung einer Aufgabendefinition verwenden, müssen Sie den JSON-Editor verwenden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

> **Important**
>
> Wenn für Ihre Aufgaben eine IAM-Rolle zur Aufgabenausführung erforderlich ist, stellen Sie sicher, dass diese in der Aufgabendefinition angegeben ist.

Wenn Sie Ihren Workload bereitstellen, verwenden Sie den `EXTERNAL`-Starttyp, wenn Sie Ihren Service erstellen oder Ihre eigenständige Aufgabe ausführen.

Im Folgenden finden Sie eine Beispielaufgabendefinition.

Linux

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "nginx",
  "image": "public.ecr.aws/nginx/nginx:latest",
  "memory": 256,
  "cpu": 256,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "nginx"
}
```

Windows

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "windows-container",
  "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-ltsc2019",
  "memory": 256,
  "cpu": 512,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "windows-container"
}
```

Amazon ECR-Image- und Aufgabendefinition (IAM-Rolle)

Im folgenden Ausschnitt wird ein Amazon-ECR-Image namens `aws-nodejs-sample` mit dem Tag `v1` von der `123456789012.dkr.ecr.us-west-2.amazonaws.com`-Registrierung verwendet. Der Container in dieser Aufgabe erbt die IAM-Berechtigungen von der Rolle `arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole`. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

```
{
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "123456789012.dkr.ecr.us-west-2.amazonaws.com/aws-nodejs-
sample:v1",
            "memory": 200,
            "cpu": 10,
            "essential": true
        }
    ],
```

```
    "family": "example_task_3",
    "taskRoleArn": "arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole"
}
```

Einstiegspunkt mit Befehl

Im folgenden Ausschnitt wird die Syntax eines Docker-Containers gezeigt, der einen Eintrittspunkt und ein Befehlsargument verwendet. Dieser Container führt viermal einen Ping für `google.com` aus und wird dann beendet.

```
{
    "containerDefinitions": [
        {
            "memory": 32,
            "essential": true,
            "entryPoint": ["ping"],
            "name": "alpine_ping",
            "readonlyRootFilesystem": true,
            "image": "alpine:3.4",
            "command": [
                "-c",
                "4",
                "example.com"
            ],
            "cpu": 16
        }
    ],
    "family": "example_task_2"
}
```

Container-Abhängigkeit

Dieser Ausschnitt veranschaulicht die Syntax einer Aufgabendefinition mit mehreren Containern, für die eine Container-Abhängigkeit angegeben ist. In der folgenden Aufgabendefinition muss der `envoy`-Container einen fehlerfreien Status erreichen. Dieser wird anhand der erforderlichen Container-Parameter für die Zustandsprüfung bestimmt, bevor der `app`-Container gestartet wird. Weitere Informationen finden Sie unter Container-Abhängigkeit.

```
{
  "family": "appmesh-gateway",
  "runtimePlatform": {
```

```
        "operatingSystemFamily": "LINUX"
    },
    "proxyConfiguration":{
        "type": "APPMESH",
        "containerName": "envoy",
        "properties": [
            {
                "name": "IgnoredUID",
                "value": "1337"
            },
            {
                "name": "ProxyIngressPort",
                "value": "15000"
            },
            {
                "name": "ProxyEgressPort",
                "value": "15001"
            },
            {
                "name": "AppPorts",
                "value": "9080"
            },
            {
                "name": "EgressIgnoredIPs",
                "value": "169.254.170.2,169.254.169.254"
            }
        ]
    },
    "containerDefinitions": [
      {
        "name": "app",
        "image": "application_image",
        "portMappings": [
          {
            "containerPort": 9080,
            "hostPort": 9080,
            "protocol": "tcp"
          }
        ],
        "essential": true,
        "dependsOn": [
          {
            "containerName": "envoy",
            "condition": "HEALTHY"
```

```
        }
      ]
    },
    {
      "name": "envoy",
      "image": "840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/meshName/virtualNode/virtualNodeName"
        },
        {
          "name": "ENVOY_LOG_LEVEL",
          "value": "info"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "echo hello"
        ],
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
    }
  ],
  "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Beispiele für Windows-Aufgabendefinitionen

Nachfolgend finden Sie eine exemplarische Aufgabendefinition, die Ihnen den Einstieg in die Verwendung von Windows-Containern auf Amazon ECS erleichtert.

Example Amazon-ECS-Beispielkonsolenanwendung für Windows

Die folgende Aufgabendefinition ist die Amazon-ECS-Beispielkonsolenanwendung, die im zuerst ausgeführten Assistenten für Amazon ECS erzeugt wird. Sie wurde portiert, um das `microsoft/iis` Windows-Container-Image zu verwenden.

```
{
  "family": "windows-simple-iis",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "entryPoint":["powershell", "-Command"],
      "command":["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file -
Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "memory": 1024,
      "essential": true
    }
  ],
  "networkMode": "awsvpc",
  "memory": "1024",
  "cpu": "1024"
}
```

Amazon-ECS-Cluster

Ein Amazon-ECS-Cluster ist eine logische Gruppierung von Aufgaben oder Services. Zusätzlich zu Aufgaben und Services besteht ein Cluster aus den folgenden Ressourcen:

- Die Infrastrukturkapazität, die eine Kombination der folgenden sein kann:
  - Amazon EC2 EC2-Instanzen in der Cloud AWS
  - Serverlos (AWS Fargate (Fargate)) in der Cloud AWS
  - On-Premises-virtuelle-Maschinen (VM) oder -Server
- Das Netzwerk (VPC und Subnetz), in dem Ihre Aufgaben und Services ausgeführt werden

  Wenn Sie Amazon-EC2-Instances für die Kapazität verwenden, kann sich das Subnetz in Availability Zones, Local Zones, Wavelength Zones oder AWS Outposts befinden.
- Ein optionaler Namespace

  Der Namespace wird für die service-to-service Kommunikation mit Service Connect verwendet.
- Eine Überwachungsoptionen

  CloudWatch Container Insights ist mit zusätzlichen Kosten verbunden und ist ein vollständig verwalteter Service. Sammelt, aggregiert und fasst Amazon-ECS-Metriken und -Protokolle automatisch zusammen.

Im Folgenden werden allgemeine Konzepte zu Amazon-ECS-Clustern vorgestellt.

- Amazon ECS erstellt einen Standard-Cluster. Sie können zusätzliche Cluster erstellen, um Ihre Ressourcen zu trennen.
- Cluster sind AWS-Region spezifisch.
- Cluster können sich in einem der folgenden Zustände befinden.

  ACTIVE

  Der Cluster ist bereit, Aufgaben zu akzeptieren, und Sie können gegebenenfalls Container-Instances beim Cluster registrieren.

  PROVISIONING

  Dem Cluster sind Kapazitätsanbieter zugeordnet, und die Ressourcen, die für den Kapazitätsanbieter benötigt werden, werden erstellt.

DEPROVISIONING

Dem Cluster sind Kapazitätsanbieter zugeordnet, und die Ressourcen, die für den Kapazitätsanbieter benötigt werden, werden gelöscht.

FEHLGESCHLAGEN

Dem Cluster sind Kapazitätsanbieter zugeordnet, und die Ressourcen, die für den Kapazitätsanbieter benötigt werden, konnten nicht erstellt werden.

INACTIVE

Der Cluster wurde gelöscht. Cluster mit dem Status `INACTIVE` Status können für einen bestimmten Zeitraum im Konto erkennbar bleiben. Dieses Verhalten kann sich in future ändern, also stellen Sie sicher, dass Sie sich nicht darauf verlassen, dass `INACTIVE` Cluster bestehen bleiben.

- Ein Cluster kann eine Mischung aus Aufgaben enthalten, die auf AWS Fargate Amazon EC2 EC2-Instances oder externen Instances gehostet werden. Aufgaben können auf Fargate- oder EC2-Infrastrukturen als Starttyp oder als Kapazitätsanbieter-Strategie ausgeführt werden. Wenn Sie EC2 als Starttyp verwenden, verfolgt und skaliert Amazon ECS die Kapazität von Amazon EC2 Auto Scaling Scaling-Gruppen nicht. Weitere Informationen zu Starttypen finden Sie unter Amazon-ECS-Starttypen.
- Ein Cluster kann eine Mischung aus Kapazitätsanbietern für Auto-Scaling-Gruppen und Fargate-Kapazitätsanbietern enthalten. Eine Kapazitätsanbieterstrategie kann nur Auto Scaling Scaling-Gruppenkapazitätsanbieter oder Fargate-Kapazitätsanbieter umfassen.
- Sie können verschiedene Instance-Typen für den EC2-Starttyp oder die Auto Scaling Scaling-Gruppenkapazitätsanbieter verwenden. Eine Instance kann jeweils nur für einen Cluster registriert werden.
- Sie können den Zugriff auf Cluster einschränken, indem Sie benutzerdefinierte IAM-Richtlinien erstellen. Weitere Informationen finden Sie im Beispiele für Amazon ECS-Cluster Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service Abschnitt unter.
- Sie können Service Auto Scaling verwenden, um Fargate-Aufgaben zu skalieren. Weitere Informationen finden Sie unter Skalieren Sie Ihren Amazon ECS-Service automatisch.
- Sie können einen standardmäßigen Service Connect-Namespace für einen Cluster konfigurieren. Nachdem Sie einen standardmäßigen Service-Connect-Namespace festgelegt haben, können alle neuen Services, die im Cluster erstellt wurden, als Client-Services im Namespace hinzugefügt werden, indem Sie Service Connect aktivieren. Es ist keine zusätzliche Konfiguration erforderlich.

Weitere Informationen finden Sie unter Verwenden Sie Service Connect, um Amazon ECS-Services mit Kurznamen zu verbinden.

Amazon ECS-Cluster für den Starttyp Fargate

Amazon-ECS-Kapazitätsanbieter verwalten die Skalierung der Infrastruktur für Aufgaben in Ihren Clustern. Jeder Cluster kann über einen oder mehrere Kapazitätsanbieter und eine optionale Kapazitätsanbieter-Standardstrategie verfügen. Die Kapazitätsanbieterstrategie legt fest, wie die Aufgaben über die Kapazitätsanbieter eines Clusters verteilt werden. Wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen, können Sie entweder die Kapazitätsanbieter-Standardstrategie des Clusters verwenden oder eine Strategie für Kapazitätsanbieter angeben, die die Standardstrategie überschreibt.

Wenn Sie Ihre Aufgaben am ausführen AWS Fargate, müssen Sie die Kapazität nicht erstellen oder verwalten. Sie müssen dem Cluster lediglich einen der folgenden vordefinierten Kapazitätsanbieter zuordnen:

- Fargate
- Fargate Spot

Mit Amazon ECS on AWS Fargate Capacity Providers können Sie sowohl Fargate- als auch Fargate Spot-Kapazitäten für Ihre Amazon ECS-Aufgaben nutzen.

Mit Fargate Spot können Sie unterbrechungstolerante Amazon-ECS-Aufgaben zu einer im Vergleich zum Fargate-Preis ermäßigten Gebühr ausführen. Fargate Spot führt Aufgaben über freie Rechenkapazität aus. Wenn die Kapazität wieder AWS benötigt wird, werden Ihre Aufgaben mit einer zweiminütigen Warnung unterbrochen. Fargate Spot unterstützt nur Linux-Aufgaben mit der X86_64-Architektur auf Plattformversion 1.3.0 oder höher.

Wenn Aufgaben, die die Kapazitätsanbieter Fargate und Fargate Spot verwenden, gestoppt werden, wird das Ereignis zur Änderung des Aufgabenstatus an Amazon gesendet. EventBridge Der Grund für das Anhalten gibt eine Beschreibung der Ursache an. Weitere Informationen finden Sie unter Ereignisse zur Änderung des Amazon ECS-Aufgabenstatus.

Ein Cluster kann eine Mischung aus Fargate- und Auto-Scaling-Gruppen-Kapazitätsanbietern enthalten. Eine Kapazitätsanbieter-Strategie kann jedoch nur entweder Fargate- oder Auto-Scaling-Gruppen-Kapazitätsanbieter enthalten, aber nicht beide. Weitere Informationen finden Sie unter Auto Scaling Group Capacity Providers.

Berücksichtigen Sie bei der Verwendung von Kapazitätsanbietern die folgenden Punkte:

- Sie müssen einem Cluster einen Kapazitätsanbieter zuordnen, bevor Sie ihn der Kapazitätsanbieterstrategie zuordnen können.
- Sie können maximal 20 Kapazitätsanbieter für eine Kapazitätsanbieterstrategie angeben.
- Sie können einen Service, der einen Kapazitätsanbieter einer Auto-Scaling-Gruppe verwendet, nicht aktualisieren, um einen Fargate-Kapazitätsanbieter zu verwenden. Das Gegenteil ist der Fall.
- Wenn in einer Kapazitätsanbieter-Strategie kein `weight`-Wert für einen Kapazitätsanbieter in der Konsole angegeben ist, wird der Standardwert von `1` verwendet. Wenn Sie die API oder verwenden AWS CLI, `0` wird der Standardwert von verwendet.
- Wenn in einer Kapazitätsanbieter-Strategie mehrere Kapazitätsanbieter angegeben sind, muss mindestens einer der Kapazitätsanbieter eine Gewichtung haben, die größer ist als Null. Kapazitätsanbieter mit einer Gewichtung von Null werden nicht für die Vergabe von Aufgaben verwendet. Wenn Sie in einer Strategie mehrere Kapazitätsanbieter mit derselben Gewichtung von Null angeben, schlagen alle `RunTask`- oder `CreateService`-Aktionen, die die Kapazitätsanbieter-Strategie verwenden, fehl.
- In einer Kapazitätsanbieter-Strategie kann nur für einen Kapazitätsanbieter ein Basis-Wert festgelegt werden. Wenn kein Basiswert angegeben wird, wird der Standardwert Null verwendet.
- Ein Cluster kann eine Mischung aus Kapazitätsanbietern für Auto-Scaling-Gruppen und Fargate-Kapazitätsanbietern enthalten. Eine Kapazitätsanbieter-Strategie kann jedoch nur Kapazitätsanbieter der Auto-Scaling-Gruppe oder Fargate enthalten, aber nicht beides.
- Ein Cluster kann eine Mischung aus Services und eigenständigen Aufgaben enthalten, die sowohl Kapazitätsanbieter als auch Starttypen verwenden. Ein Service kann aktualisiert werden, um anstelle eines Starttyps eine Kapazitätsanbieterstrategie zu verwenden. Allerdings müssen Sie dabei eine neue Bereitstellung erzwingen.

Kündigungsmitteilungen von Fargate Spot

In Zeiten extrem hoher Nachfrage sind die Kapazitäten von Fargate Spot möglicherweise nicht verfügbar. Dies kann dazu führen, dass Fargate-Spot-Aufgaben verzögert werden. In diesem Fall versuchen die Amazon ECS-Services erneut, Aufgaben zu starten, bis die erforderliche Kapazität verfügbar ist. Fargate ersetzt Spot-Kapazität nicht durch On-Demand-Kapazität.

Wenn Aufgaben, die Fargate Spot-Kapazität verwenden, aufgrund einer Spot-Unterbrechung angehalten werden, wird zwei Minuten, bevor eine Aufgabe angehalten wird, eine Warnung gesendet.

Die Warnung wird als Ereignis zur Änderung des Aufgabenstatus an Amazon EventBridge und als SIGTERM-Signal an die laufende Aufgabe gesendet. Wenn Sie Fargate Spot als Teil eines Service verwenden, empfängt der Service-Planer in diesem Szenario das Unterbrechungssignal und versucht, zusätzliche Aufgaben auf Fargate Spot zu starten, wenn Kapazität verfügbar ist. Ein Service mit nur einer Aufgabe wird unterbrochen, bis Kapazität verfügbar ist. Weitere Informationen zu einem korrekten Herunterfahren finden Sie unter Korrektes Herunterfahren mit ECS.

Um sicherzustellen, dass Ihre Container korrekt beendet werden, bevor die Aufgabe endet, können Sie Folgendes konfigurieren:

- In der Containerdefinition, die die Aufgabe verwendet, kann ein `stopTimeout`-Wert von `120` Sekunden oder weniger angegeben werden. Der `stopTimeout`-Standardwert liegt bei 30 Sekunden. Sie können einen längeren `stopTimeout`-Wert angeben, um mehr Zeit zwischen dem Eingang des Ereignisses zur Änderung des Aufgabenstatus und dem Zeitpunkt, an dem der Container zwangsweise angehalten wird, zu gewinnen. Weitere Informationen finden Sie unter Container-Timeouts.
- Das SIGTERM-Signal muss innerhalb des Containers empfangen werden, um alle Bereinigungsaktionen ausführen zu können. Wird dieses Signal nicht verarbeitet, erhält die Aufgabe nach dem konfigurierten `stopTimeout` ein SIGKILL-Signal, was zu Datenverlust oder -beschädigung führen kann.

Das Folgende ist ein Ausschnitt eines Ereignisses zur Änderung des Aufgabenstatus. Dieser Codeausschnitt zeigt den Anhaltegrund und den Anhaltecode für eine Fargate-Spot-Unterbrechung an.

```
{
  "version": "0",
  "id": "9bcdac79-b31f-4d3d-9410-fbd727c29fab",
  "detail-type": "ECS Task State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:task/b99d40b3-5176-4f71-9a52-9dbd6f1cebef"
  ],
  "detail": {
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "createdAt": "2016-12-06T16:41:05.702Z",
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
```

```
    "stoppedReason": "Your Spot Task was interrupted.",
    "stopCode": "SpotInterruption",
    "taskArn": "arn:aws:ecs:us-east-1:111122223333:task/
b99d40b3-5176-4f71-9a52-9dbd6fEXAMPLE",
    ...
  }
}
```

Das Folgende ist ein Ereignismuster, das verwendet wird, um eine EventBridge Regel für Ereignisse zur Änderung des Amazon ECS-Aufgabenstatus zu erstellen. Optional können Sie einen Cluster im `detail`-Feld angeben. Das bedeutet, dass Sie Ereignisse zur Statusänderung von Aufgaben für diesen Cluster erhalten. Weitere Informationen finden Sie unter EventBridge Regel erstellen im EventBridge Amazon-Benutzerhandbuch.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Task State Change"
    ],
    "detail": {
        "clusterArn": [
            "arn:aws:ecs:us-west-2:111122223333:cluster/default"
        ]
    }
}
```

Erstellen eines Amazon ECS-Clusters für den Starttyp Fargate

Sie können mit der Amazon ECS-Konsole einen Amazon ECS-Cluster erstellen. Bevor Sie beginnen, vergewissern Sie sich, dass Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS ausgeführt haben, und weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”. Die Amazon ECS-Konsole erstellt die Ressourcen, die von einem Amazon ECS-Cluster benötigt werden, indem sie einen AWS CloudFormation Stack erstellt.

Die Konsole ordnet dem Cluster automatisch die Kapazitätsanbieter Fargate und Fargate Spot zu.

Zusätzlich zum Cluster erstellt die Konsole automatisch die folgenden Ressourcen:

- Ein Standard-Namespace AWS Cloud Map , der denselben Namen wie der Cluster hat. Ein Namespace ermöglicht es Services, die Sie im Cluster erstellen, ohne zusätzliche Konfiguration eine Verbindung zu den anderen Services im Namespace herzustellen.

  Weitere Informationen finden Sie unter Amazon ECS-Services verbinden.

Sie können die folgenden Optionen ändern:

- Ändern Sie den Standard-Namespace, der dem Cluster zugeordnet ist.
- Aktivieren von Container Insights.

  CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Container Insights bietet auch Diagnoseinformationen, wie z. B. Fehler beim Container-Neustart, damit Sie Probleme schnell isolieren und beheben können. Weitere Informationen finden Sie unter the section called “Überwachen Sie Amazon ECS-Container mit Container Insights”.
- Fügen Sie Tags hinzu, die Ihnen helfen, Ihren Cluster zu identifizieren.

Verfahren

So erstellen Sie einen neuen Cluster (Amazon ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Konfigurieren Sie unter Clusterkonfiguration Folgendes:

   - Geben Sie für Cluster-Name einen eindeutigen Namen ein.

     Der Name kann bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Bindestriche enthalten.
   - (Optional) Wenn der für Service Connect verwendete Namespace nicht mit dem Clusternamen identisch sein soll, geben Sie unter Namespace einen eindeutigen Namen ein.
6. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).

7. (Optional) Um Ihren Cluster leichter identifizieren zu können, erweitern Sie den Bereich Tags, und konfigurieren Sie dann Ihre Tags.

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

   [Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.
8. Wählen Sie Erstellen.

Nächste Schritte

Nachdem Sie den Cluster erstellt haben, können Sie Aufgabendefinitionen für Ihre Anwendungen erstellen und diese dann als eigenständige Aufgaben oder als Teil eines Services ausführen. Weitere Informationen finden Sie hier:

- Amazon-ECS-Aufgabendefinitionen
- Eine Anwendung als Amazon ECS-Aufgabe ausführen
- Einen Amazon ECS-Service mithilfe der Konsole erstellen

Amazon ECS-Kapazitätsanbieter für den EC2-Starttyp

Wenn Sie Amazon-EC2-Instances für Ihre Kapazität nutzen, verwenden Sie Auto-Scaling-Gruppen, um die Amazon-EC2-Instances zu verwalten, die in ihren Clustern registriert sind. Auto Scaling hilft sicherzustellen, dass Ihnen die richtige Anzahl von Amazon EC2 EC2-Instances zur Verfügung steht, um die Anwendungslast zu bewältigen.

Sie können die Funktion für verwaltete Skalierung verwenden, damit Amazon ECS die Scale-In- und Scale-Out-Aktionen der Auto Scaling Scaling-Gruppe verwaltet, oder Sie können die Skalierungsaktionen selbst verwalten. Weitere Informationen finden Sie unter Automatische Verwaltung der Amazon ECS-Kapazität mit Cluster-Auto-Scaling.

Wir empfehlen, dass Sie eine neue leere Auto Scaling Scaling-Gruppe erstellen. Bei Verwendung einer vorhandenen Auto-Scaling-Gruppe kann es vorkommen, dass Amazon-EC2-Instances,

die der Gruppe zugeordnet sind und die bereits ausgeführt und bei einem Amazon-ECS-Cluster registriert waren, bevor die Auto-Scaling-Gruppe zur Erstellung eines Kapazitätsanbieters verwendet wurde, nicht ordnungsgemäß bei dem Kapazitätsanbieter registriert werden. Dies kann Probleme verursachen, wenn der Kapazitätsanbieter in einer Kapazitätsanbieter-Strategie verwendet wird. Verwenden Sie `DescribeContainerInstances`, um zu bestätigen, ob eine Container-Instance einem Kapazitätsanbieter zugeordnet ist oder nicht.

> **Note**
>
> Um eine leere Auto-Scaling-Gruppe zu erstellen, setzen Sie die gewünschte Anzahl auf Null. Nachdem Sie den Kapazitätsanbieter erstellt und einem Cluster zugeordnet haben, können Sie ihn aufskalieren.
> Wenn Sie die Amazon ECS-Konsole verwenden, erstellt Amazon ECS in Ihrem Namen eine Amazon EC2 EC2-Startvorlage und eine Auto Scaling Scaling-Gruppe als Teil des AWS CloudFormation Stacks. Ihnen wird das Präfix vorangestellt. `EC2ContainerService-<ClusterName>` Sie können die Auto-Scaling-Gruppe als Kapazitätsanbieter für diesen Cluster verwenden.

Wir empfehlen Ihnen, Managed Instance Draining zu verwenden, um Amazon EC2 EC2-Instances ordnungsgemäß zu beenden, ohne Ihre Workloads zu stören. Diese Funktion ist standardmäßig aktiviert. Weitere Informationen finden Sie unter Beenden Sie sicher Amazon ECS-Workloads, die auf EC2-Instances ausgeführt werden.

Beachten Sie Folgendes, wenn Sie die Kapazitätsanbieter der Auto-Scaling-Gruppe in der Konsole verwenden:

- Eine Auto-Scaling-Gruppe muss eine `MaxSize` größer als Null haben, um eine Aufskalierung zu ermöglichen.
- Die Auto-Scaling-Gruppe kann keine Einstellungen für die Instance-Gewichtung haben.
- Wenn die Auto-Scaling-Gruppe nicht aufskaliert werden kann, um die Anzahl der ausgeführten Aufgaben aufzunehmen, können die Aufgaben nicht über den `PROVISIONING`-Status hinausgehen.
- Ändern Sie nicht die Ressource der Skalierungsrichtlinie, die Ihren Auto-Scaling-Gruppen zugeordnet ist, die von Kapazitätsanbietern verwaltet werden.
- Wenn die verwaltete Skalierung beim Erstellen eines Kapazitätsanbieters aktiviert ist, kann die gewünschte Anzahl der Auto-Scaling-Gruppe auf `0` festgelegt werden. Wenn die verwaltete

Skalierung aktiviert ist, verwaltet Amazon ECS die Auf- und Abskalierungsaktionen der Auto-Scaling-Gruppe.

- Sie müssen einen Kapazitätsanbieter einem Cluster zuordnen, bevor Sie ihn der Kapazitätsanbieter-Strategie zuordnen können.
- Sie können maximal 20 Kapazitätsanbieter für eine Kapazitätsanbieterstrategie angeben.
- Sie können einen Service, der einen Kapazitätsanbieter einer Auto-Scaling-Gruppe verwendet, nicht aktualisieren, um einen Fargate-Kapazitätsanbieter zu verwenden. Das Gegenteil ist der Fall.
- Wenn in einer Kapazitätsanbieter-Strategie kein `weight`-Wert für einen Kapazitätsanbieter in der Konsole angegeben ist, wird der Standardwert von 1 verwendet. Wenn Sie die API oder verwenden AWS CLI, `0` wird der Standardwert von verwendet.
- Wenn in einer Kapazitätsanbieter-Strategie mehrere Kapazitätsanbieter angegeben sind, muss mindestens einer der Kapazitätsanbieter eine Gewichtung haben, die größer ist als Null. Kapazitätsanbieter mit einer Gewichtung von Null werden nicht für die Vergabe von Aufgaben verwendet. Wenn Sie in einer Strategie mehrere Kapazitätsanbieter mit derselben Gewichtung von Null angeben, schlagen alle `RunTask`- oder `CreateService`-Aktionen, die die Kapazitätsanbieter-Strategie verwenden, fehl.
- In einer Kapazitätsanbieter-Strategie kann nur für einen Kapazitätsanbieter ein Basis-Wert festgelegt werden. Wenn kein Basiswert angegeben wird, wird der Standardwert Null verwendet.
- Ein Cluster kann eine Mischung aus Kapazitätsanbietern für Auto-Scaling-Gruppen und Fargate-Kapazitätsanbietern enthalten. Eine Kapazitätsanbieter-Strategie kann jedoch nur Kapazitätsanbieter der Auto-Scaling-Gruppe oder Fargate enthalten, aber nicht beides.
- Ein Cluster kann eine Mischung aus Services und eigenständigen Aufgaben enthalten, die sowohl Kapazitätsanbieter als auch Starttypen verwenden. Ein Service kann aktualisiert werden, um anstelle eines Starttyps eine Kapazitätsanbieterstrategie zu verwenden. Allerdings müssen Sie dabei eine neue Bereitstellung erzwingen.
- Amazon ECS unterstützt Warm-Pools für Amazon EC2 Auto Scaling. Ein Warm-Pool ist eine Gruppe von vorinitialisierten Amazon-EC2-Instances, die bereit sind, in Betrieb genommen zu werden. Wann immer Ihre Anwendung skaliert werden muss, verwendet Amazon EC2 Auto Scaling die vorinitialisierten Instances aus dem warmen Pool, anstatt kalte Instances zu starten. Dadurch kann jeder letzte Initialisierungsprozess ausgeführt werden, bevor die Instance in Betrieb genommen wird. Weitere Informationen finden Sie unter Konfiguration vorinitialisierter Instances für Ihre Amazon ECS Auto Scaling Scaling-Gruppe.

Weitere Informationen zum Erstellen einer Startvorlage für Amazon EC2 Auto Scaling finden Sie unter Startvorlagen im Benutzerhandbuch zu Amazon EC2 Auto Scaling. Weitere Informationen zum Erstellen einer Gruppe für Amazon EC2 Auto Scaling finden Sie unter Auto-Scaling-Gruppen im Benutzerhandbuch zu Amazon EC2 Auto Scaling.

Sicherheitsüberlegungen für Amazon EC2 EC2-Container-Instances für Amazon ECS

Sie sollten eine einzelne Container-Instance und ihren Zugriff innerhalb Ihres Bedrohungsmodells berücksichtigen. Beispielsweise könnte eine einzelne betroffene Aufgabe in der Lage sein, die IAM-Berechtigungen einer nicht infizierten Aufgabe auf derselben Instance zu nutzen.

Wir empfehlen Ihnen Folgendes, um dies zu verhindern:

- Verwenden Sie bei der Ausführung Ihrer Aufgaben keine Administratorrechte.
- Weisen Sie Ihren Aufgaben eine Aufgabenrolle mit der geringsten Berechtigung zu.

  Der Container-Agent erstellt automatisch ein Token mit einer eindeutigen Anmeldeinformations-ID, das für den Zugriff auf Amazon-ECS-Ressourcen verwendet wird.
- Um zu verhindern, dass Container, die von Aufgaben ausgeführt werden, die den `awsvpc`-Netzwerkmodus verwenden, auf die Anmeldeinformationen zugreifen, die dem Amazon-EC2-Instance-Profil zugewiesen wurden, während die Berechtigungen, die von der Aufgabenrolle bereitgestellt werden, weiterhin zulässig sind, setzen Sie die `ECS_AWSVPC_BLOCK_IMDS`-Agentenkonfigurationsvariable in der Agentenkonfigurationsdatei auf wahr und starten Sie den Agenten neu.
- Verwenden Sie Amazon GuardDuty Runtime Monitoring, um Bedrohungen für Cluster und Container in Ihrer AWS Umgebung zu erkennen. Runtime Monitoring verwendet einen GuardDuty Sicherheitsagenten, der die Laufzeit einzelner Amazon ECS-Workloads transparent macht, z. B. Dateizugriff, Prozessausführung und Netzwerkverbindungen. Weitere Informationen finden Sie unter GuardDutyRuntime Monitoring im GuardDuty Benutzerhandbuch.

Erstellen eines Amazon ECS-Clusters für den Amazon EC2 EC2-Starttyp

Sie können mit der Konsole einen Amazon ECS-Cluster erstellen. Bevor Sie beginnen, vergewissern Sie sich, dass Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS ausgeführt haben, und weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”. Die Amazon ECS-Konsole bietet eine einfache

Möglichkeit, die Ressourcen zu erstellen, die von einem Amazon ECS-Cluster benötigt werden, indem ein AWS CloudFormation Stack erstellt wird.

Um den Prozess der Clustererstellung so einfach wie möglich zu gestalten, verfügt die Konsole über Standardauswahlen für viele Auswahlmöglichkeiten, die wir unten beschreiben. Es gibt auch Hilfe-Panels für die meisten Abschnitte in der Konsole, die weiteren Kontext bieten.

Sie können Amazon-EC2-Instances registrieren, wenn Sie den Cluster erstellen, oder zusätzliche Instances mit dem Cluster registrieren, nachdem er erstellt wurde.

Sie können die folgenden Standardoptionen ändern:

- Subnetze ändern, in denen Ihre Instances gestartet werden
- Sicherheitsgruppen ändern, die zur Steuerung des Datenverkehrs zu den Container-Instances verwendet werden
- Ändern Sie den Standard-Namespace, der dem Cluster zugeordnet ist.

  Über einen Namespace können Services, die Sie im Cluster erstellen, ohne zusätzliche Konfiguration eine Verbindung zu den anderen Services im Namespace herstellen. Der Standard-Namespace ist mit dem Cluster-Namen identisch. Weitere Informationen finden Sie unter Amazon ECS-Services verbinden.
- Aktivieren von Container Insights.

  CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Container Insights bietet auch Diagnoseinformationen, wie z. B. Fehler beim Container-Neustart, damit Sie Probleme schnell isolieren und beheben können. Weitere Informationen finden Sie unter the section called “Überwachen Sie Amazon ECS-Container mit Container Insights”.
- Fügen Sie Tags hinzu, die Ihnen helfen, Ihren Cluster zu identifizieren.

Optionen für Auto-Scaling-Gruppen

Wenn Sie Amazon-EC2-Instances verwenden, müssen Sie eine Auto-Scaling-Gruppe angeben, um die Infrastruktur zu verwalten, auf der Ihre Aufgaben und Services ausgeführt werden.

Wenn Sie eine neue Auto-Scaling-Gruppe erstellen möchten, wird diese automatisch für das folgende Verhalten konfiguriert:

- Die Aktionen der Auto-Scaling-Gruppe zum Ab- oder Aufskalieren werden von Amazon ECS verwaltet.
- Amazon ECS verhindert nicht, dass Amazon-EC2-Instances, die Aufgaben enthalten und sich in einer Auto-Scaling-Gruppe befinden, während einer Abskalierungs-Aktion beendet werden. Weitere Informationen finden Sie unter Instance-Skalierungsschutz im AWS Auto Scaling - Benutzerhandbuch.

Sie konfigurieren die folgenden Eigenschaften für Auto-Scaling-Gruppen. Diese bestimmen den Typ und die Anzahl der Instances, die für die Gruppe gelauncht werden sollen:

- Das Amazon-ECS-optimierte AMI.
- Der Instance-Typ.
- Das SSH-Schlüsselpaar, das Ihre Identität beweist, wenn Sie eine Verbindung mit der Instance herstellen. Informationen zur Erstellung von SSH-Schlüsseln finden Sie unter Amazon EC2 EC2-Schlüsselpaare und Linux-Instances im Amazon EC2 EC2-Benutzerhandbuch.
- Die minimale Anzahl von Instances, die für die Auto-Scaling-Gruppe gelauncht werden sollen.
- Die maximale Anzahl von Instances, die für die Auto-Scaling-Gruppe gestartet werden.

  Damit die Gruppe aufskaliert werden kann, muss das Maximum größer als 0 sein.

Amazon ECS erstellt als Teil des AWS CloudFormation -Stacks in Ihrem Namen eine Launchvorlage von Amazon EC2 Auto Scaling und eine Auto-Scaling-Gruppe. Die Werte, die Sie für das AMI, die Instance-Typen und das SSH-Schlüsselpaar angegeben haben, befinden sich in der Startvorlage. Die Vorlagen sind mit dem Präfix `EC2ContainerService-<ClusterName>` versehen, wodurch sie leicht erkennbar sind. Die Auto-Scaling-Gruppen haben das Präfix `<ClusterName>-ECS-Infra-ECSAutoScalingGroup`.

Instances, die für die Auto-Scaling-Gruppe geluncht wurden, verwenden die Launchvorlage.

Netzwerkoptionen

Standardmäßig werden Instances in den Standard-Subnetzen für die Region gestartet. Die Sicherheitsgruppen, die den Verkehr zu Ihren Container-Instances steuern, werden verwendet, die derzeit den Subnetzen zugeordnet sind. Sie können die Subnetze und Sicherheitsgruppen für die Instances ändern.

Sie können ein vorhandenes Subnetz auswählen. Sie können entweder eine bestehende Sicherheitsgruppe verwenden oder eine neue erstellen. Wenn Sie eine neue Sicherheitsgruppe erstellen, müssen Sie mindestens eine Regel für eingehenden Datenverkehr angeben.

Die Regeln für eingehenden Datenverkehr legen fest, welcher Datenverkehr Ihre Container-Instances erreichen kann, und beinhalten die folgenden Eigenschaften:

- Das zulässige Protokoll
- Der zulässige Bereich an Ports
- Der eingehende Verkehr (Quelle)

Um eingehenden Verkehr von einer bestimmten Adresse oder einem bestimmten CIDR-Block zuzulassen, verwenden Sie Benutzerdefiniert als Quelle mit dem zulässigen CIDR.

Um eingehenden Datenverkehr von allen Zielen zuzulassen, verwenden Sie Überall als Quelle. Dies fügt automatisch den IPv4-CIDR-Block 0.0.0.0/0 und den IPv6-CIDR-Block ::/0 hinzu.

Um eingehenden Datenverkehr von Ihrem lokalen Computer zuzulassen, verwenden Sie Quellengruppe als Quelle. Dadurch wird automatisch die aktuelle IP-Adresse Ihres lokalen Computers als zulässige Quelle hinzugefügt.

So erstellen Sie einen neuen Cluster (Amazon ECS-Konsole)

Bevor Sie beginnen, weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Konfigurieren Sie unter Clusterkonfiguration Folgendes:

    - Geben Sie für Cluster-Name einen eindeutigen Namen ein.

      Der Name kann bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Bindestriche enthalten.
    - (Optional) Wenn der für Service Connect verwendete Namespace nicht mit dem Clusternamen identisch sein soll, geben Sie unter Namespace einen eindeutigen Namen ein.

6. Fügen Sie Ihrem Cluster Amazon-EC2-Instances hinzu, erweitern Sie Infrastruktur, entfernen Sie AWS Fargate (Serverless) und wählen Sie dann Amazon-EC2-Instances. Konfigurieren Sie als Nächstes die Auto-Scaling-Gruppe, die als Kapazitätsanbieter fungiert:

   a. Um eine vorhandene Auto-Scaling-Gruppe zu verwenden, wählen Sie die Gruppe aus der Auto-Scaling-Gruppe (ASG) aus.

   b. Um eine Auto-Scaling-Gruppe zu erstellen, wählen Sie in der Auto-Scaling-Gruppe (ASG) Create new group (Neue Gruppe erstellen) und geben Sie dann die folgenden Details zur Gruppe an:

      - Wählen Sie für das Bereitstellungsmodell aus, ob On-Demand-Instances oder Spot-Instances verwendet werden sollen.
      - Wenn Sie Spot-Instances verwenden möchten, wählen Sie für die Zuweisungsstrategie aus, welche Spot-Kapazitätspools (Instance-Typen und Availability Zones) für die Instances verwendet werden.

        Für die meisten Workloads können Sie „Preis und Kapazität optimiert“ wählen.

        Weitere Informationen finden Sie unter Zuweisungsstrategien für Spot-Instances im Amazon-EC2-Benutzerhandbuch.
      - Wählen Sie für Operating system/Architecture (Betriebssystem/Architektur) das Amazon-ECS-optimierte AMI für die Auto-Scaling-Gruppen-Instances aus.
      - Wählen Sie für EC2 instance type (EC2-Instance-Typ) den Instance-Typ für Ihre Workloads aus.

        Die verwaltete Skalierung funktioniert am besten, wenn Ihre Auto-Scaling-Gruppe dieselben oder ähnliche Instance-Typen verwendet.
      - Wählen Sie für die EC2-Instance-Rolle eine vorhandene Container-Instance-Rolle aus, oder Sie können eine neue erstellen.

        Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.
      - Geben Sie für Capacity (Kapazität) die minimale Anzahl und die maximale Anzahl von Instances ein, die in der Auto-Scaling-Gruppe gelauncht werden sollen.
      - Wählen Sie für SSH key pair (SSH-Schlüsselpaar) das Paar aus, das Ihre Identität nachweist, wenn Sie eine Verbindung zur Instance herstellen.
      - Um ein größeres Bild und größeren Speicherplatz zu ermöglichen, geben Sie für die Größe des Root-EBS-Volumes den Wert in GiB ein.

7. (Optional) Um die VPC und Subnetze zu ändern, führen Sie unter Netzwerke für Amazon-EC2-Instances einen der folgenden Vorgänge aus:

   - Um ein Subnetz zu entfernen, wählen Sie unter Subnets (Subnetze) X für jedes Subnetz, das Sie entfernen möchten.
   - Um zu einer anderen VPC als der Standard-VPC zu wechseln, wählen Sie unter VPC eine vorhandene VPC und dann unter Subnetze die Subnetze aus.
   - Wählen Sie die Sicherheitsgruppen aus. Wählen Sie unter Sicherheitsgruppe eine der folgenden Optionen aus:
     - Um eine vorhandene Sicherheitsgruppe zu verwenden, wählen Sie Vorhandene Sicherheitsgruppe verwenden und dann die Sicherheitsgruppe.
     - Um eine Sicherheitsgruppe zu erstellen, wählen Sie Neue Sicherheitsgruppe erstellen aus. Wählen Sie dann Regel hinzufügen für jede eingehende Regel.

       Informationen zu Eingangsregeln finden Sie unter Netzwerkoptionen.
   - Um Ihren Amazon-EC2-Container-Instances automatisch öffentliche IP-Adressen zuzuweisen, wählen Sie für Automatische Zuweisung öffentlicher IP-Adressen eine der folgenden Optionen:
     - Subnetzeinstellung verwenden – Weisen Sie den Instances eine öffentliche IP-Adresse zu, wenn es sich bei dem Subnetz, in dem die Instances gestartet werden, um ein öffentliches Subnetz handelt.
     - Einschalten – Weisen Sie den Instances eine öffentliche IP-Adresse zu.
8. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).
9. (Optional)

   Wenn Sie Runtime Monitoring mit der manuellen Option verwenden und möchten, dass dieser Cluster von überwacht wird GuardDuty, wählen Sie Tag hinzufügen und gehen Sie wie folgt vor:

   - Geben Sie als Schlüssel ein **`guardDutyRuntimeMonitoringManaged`**
   - Geben Sie für Wert **`true`** ein.
10. (Optional) Um die Cluster-Tags zu verwalten, erweitern Sie Tags und führen Sie dann eine der folgenden Vorgänge aus:

    [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

- Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
- Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

[Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.

11. Wählen Sie Erstellen.

Nächste Schritte

Nachdem Sie den Cluster erstellt haben, können Sie Aufgabendefinitionen für Ihre Anwendungen erstellen und diese dann als eigenständige Aufgaben oder als Teil eines Services ausführen. Weitere Informationen finden Sie hier:

- Amazon-ECS-Aufgabendefinitionen
- Eine Anwendung als Amazon ECS-Aufgabe ausführen
- Einen Amazon ECS-Service mithilfe der Konsole erstellen

Automatische Verwaltung der Amazon ECS-Kapazität mit Cluster-Auto-Scaling

Amazon ECS kann die Skalierung von Amazon-EC2-Instances verwalten, die in Ihrem Cluster registriert sind. Dies wird als Auto Scaling für Amazon-ECS-Cluster bezeichnet. Sie aktivieren die verwaltete Skalierung, wenn Sie den Amazon ECS Auto Scaling Scaling-Gruppenkapazitätsanbieter erstellen. Anschließend legen Sie einen Zielprozentsatz (`dentargetCapacity`) für die Instanznutzung in dieser Auto Scaling Scaling-Gruppe fest. Amazon ECS erstellt zwei benutzerdefinierte CloudWatch Metriken und eine Skalierungsrichtlinie für die Zielverfolgung für Ihre Auto Scaling Scaling-Gruppe. Amazon ECS verwaltet dann die Scale-In- und Scale-Out-Aktionen auf der Grundlage der Ressourcenauslastung, die Ihre Aufgaben verbrauchen.

Für jeden Kapazitätsanbieter der Auto-Scaling-Gruppe, der einem Cluster zugeordnet ist, erstellt und verwaltet Amazon ECS die folgenden Ressourcen:

- Ein Alarm bei niedrigem metrischen Wert CloudWatch
- Ein CloudWatch Alarm bei hohem metrischen Wert
- Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung.

> **Note**
>
> Amazon ECS erstellt die Skalierungsrichtlinie für die Ziel-Nachverfolgung und fügt sie an die Auto-Scaling-Gruppe an. Um die Skalierungsrichtlinie für die Ziel-Nachverfolgung zu aktualisieren, aktualisieren Sie die vom Kapazitätsanbieter verwalteten Skalierungseinstellungen, statt die Skalierungsrichtlinie direkt zu aktualisieren.

Wenn Sie die verwaltete Skalierung deaktivieren oder den Kapazitätsanbieter von einem Cluster trennen, entfernt Amazon ECS sowohl die CloudWatch Metriken als auch die Ressourcen der Ziel-Tracking-Skalierungsrichtlinie.

Amazon ECS verwendet die folgenden Metriken, um zu bestimmen, welche Maßnahmen zu ergreifen sind:

`CapacityProviderReservation`

Der Prozentsatz der Container-Instances, die für einen bestimmten Kapazitätsanbieter verwendet werden. Amazon ECS generiert diese Metrik.

Amazon ECS legt den Wert `CapacityProviderReservation` auf eine Zahl zwischen 0–100 fest. Amazon ECS verwendet die folgende Formel, um das Verhältnis der verbleibenden Kapazität in der Auto-Scaling-Gruppe darzustellen. Anschließend veröffentlicht Amazon ECS die Metrik für CloudWatch. Weitere Informationen zur Berechnung der Metrik finden Sie unter Deep Dive on Amazon ECS Cluster Auto Scaling.

```
CapacityProviderReservation = (number of instances needed) / (number of running
 instances) x 100
```

`DesiredCapacity`

Die Kapazitätsmenge für die Auto-Scaling-Gruppe. Diese Metrik wurde nicht veröffentlicht CloudWatch.

Amazon ECS veröffentlicht die `CapacityProviderReservation` Metrik CloudWatch im `AWS/ECS/ManagedScaling` Namespace. Die `CapacityProviderReservation`-Metrik führt zu einer der folgenden Aktionen:

Der Wert **`CapacityProviderReservation`** entspricht **`targetCapacity`**

Die Auto-Scaling-Gruppe muss weder ab- noch aufskaliert werden. Der angestrebte Nutzungsprozentsatz wurde erreicht.

Der Wert **`CapacityProviderReservation`** ist größer als **`targetCapacity`**

Es gibt mehr Aufgaben, die einen höheren Prozentsatz der Kapazität beanspruchen als Ihr `targetCapacity`-Prozentsatz. Der erhöhte Wert der `CapacityProviderReservation` Metrik führt dazu, dass der zugehörige CloudWatch Alarm ausgelöst wird. Dieser Alarm aktualisiert den `DesiredCapacity`-Wert für die Auto-Scaling-Gruppe. Die Auto-Scaling-Gruppe verwendet diesen Wert, um EC2-Instances zu launchen und sie dann beim Cluster anzumelden.

Wenn die `targetCapacity` der Standardwert 100 % ist, befinden sich die neuen Aufgaben im Status `PENDING` während der Skalierung, da auf den Instances keine Kapazität zur Ausführung der Aufgaben verfügbar ist. Nachdem sich die neuen Instances bei ECS registriert haben, werden diese Aufgaben auf den neuen Instances gestartet.

Der Wert **`CapacityProviderReservation`** ist kleiner als **`targetCapacity`**

Es gibt weniger Aufgaben, die einen niedrigeren Prozentsatz der Kapazität beanspruchen als Ihr `targetCapacity`-Prozentsatz, und es gibt mindestens eine Instance, die beendet werden kann. Der verringerte Wert der `CapacityProviderReservation` Metrik bewirkt, dass der zugehörige CloudWatch Alarm ausgelöst wird. Dieser Alarm aktualisiert den `DesiredCapacity`-Wert für die Auto-Scaling-Gruppe. Die Auto-Scaling-Gruppe verwendet diesen Wert, um EC2-Container-Instances zu beenden und sie dann vom Cluster abzumelden.

Die Auto-Scaling-Gruppe folgt der Beendigungsrichtlinie der Gruppe, um zu bestimmen, welche Instances sie bei Abskalieren-Ereignissen zuerst beendet. Außerdem werden Instances vermieden, für die der Instance-Abskalierungsschutz aktiviert ist. Cluster-Auto-Scaling kann verwalten, für welche Instances die Einstellung für den Instance-Abskalierungsschutz gilt, wenn Sie den verwalteten Beendigungsschutz aktivieren. Weitere Informationen zum verwalteten Beendigungsschutz finden Sie unter Steuern Sie die Instanzen, die Amazon ECS beendet. Weitere Informationen darüber, wie Auto-Scaling-Gruppen Instances beenden, finden Sie unter Steuern, welche Auto-Scaling-Instances während der Abskalierung beendet werden im Benutzerhandbuch für Amazon EC2 Auto Scaling.

Bei Verwendung von Cluster-Auto-Scaling sollte Folgendes berücksichtigt werden:

- Ändern oder verwalten Sie nicht die gewünschte Kapazität für die Auto-Scaling-Gruppe, die einem Kapazitätsanbieter zugeordnet ist, der über andere Skalierungsrichtlinien verfügt als die, die Amazon ECS verwaltet.
- Amazon ECS verwendet die `AWSServiceRoleForECS` serviceverknüpfte IAM-Rolle für die Berechtigungen, die erforderlich sind, um in Ihrem Namen AWS Auto Scaling aufzurufen. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.
- Bei der Verwendung von Kapazitätsanbietern mit Auto-Scaling-Gruppen benötigt der Benutzer, die Gruppe oder die Rolle, der bzw. die die Kapazitätsanbieter erstellt, die `autoscaling:CreateOrUpdateTags`-Berechtigung. Dies liegt daran, dass Amazon ECS an die Auto-Scaling-Gruppe ein Tag hinzufügt, wenn sie es dem Kapazitätsanbieter zuordnet.

  > **Important**
  >
  > Stellen Sie sicher, dass die von Ihnen verwendeten Tools das `AmazonECSManaged`-Tag nicht aus der Auto-Scaling-Gruppe entfernen. Wenn dieses Tag entfernt wird, kann Amazon ECS die Skalierung nicht verwalten.

- Bei der auto Clusterskalierung wird das MinimumCapacityoder MaximumCapacityfür die Gruppe nicht geändert. Damit die Gruppe horizontal skaliert werden kann, MaximumCapacitymuss der Wert für größer als Null sein.
- Wenn Auto Scaling (verwaltete Skalierung) aktiviert ist, kann ein Kapazitätsanbieter nur mit einem Cluster gleichzeitig verbunden sein. Wenn Ihr Kapazitätsanbieter die verwaltete Skalierung deaktiviert hat, können Sie ihn mehreren Clustern zuordnen.
- Wenn die verwaltete Skalierung deaktiviert ist, führt der Kapazitätsanbieter keine Auf- oder Abskalierung durch. Sie können eine Kapazitätsanbieter-Strategie verwenden, um Ihre Aufgaben zwischen Kapazitätsanbietern auszugleichen.
- Die `binpack` Strategie ist in Bezug auf die Kapazität die effizienteste Strategie.
- Wenn die Zielkapazität unter 100% liegt, muss die Platzierungsstrategie die `binpack` Strategie verwenden, ohne dass die `spread` Strategie eine höhere Ordnung als die `binpack` Strategie hat. Dadurch wird verhindert, dass der Kapazitätsanbieter eine Skalierung durchführt, bis jede Aufgabe über eine eigene Instanz verfügt oder das Limit erreicht ist.

Optimieren Sie die auto Skalierung von Amazon ECS-Clustern

Kunden, die Amazon ECS auf Amazon EC2 ausführen, können die Vorteile von Cluster Auto Scaling nutzen, um die Skalierung von Amazon EC2 Auto Scaling-Gruppen zu verwalten. Mit Cluster Auto

Scaling können Sie Amazon ECS so konfigurieren, dass Ihre Auto Scaling-Gruppe automatisch skaliert wird, sodass Sie sich ganz auf die Ausführung Ihrer Aufgaben konzentrieren können. Amazon ECS stellt sicher, dass die Auto Scaling Scaling-Gruppe nach Bedarf ein- und ausskaliert wird, ohne dass weitere Eingriffe erforderlich sind. Amazon ECS-Kapazitätsanbieter werden verwendet, um die Infrastruktur in Ihrem Cluster zu verwalten, indem sie sicherstellen, dass genügend Container-Instances vorhanden sind, um die Anforderungen Ihrer Anwendung zu erfüllen. Weitere Informationen darüber, wie Cluster-Auto-Scaling unter der Haube funktioniert, finden Sie unter Deep Dive on Amazon ECS Cluster Auto Scaling.

Cluster-Auto-Scaling basiert auf einer CloudWatch basierten Integration mit der Auto Scaling-Gruppe zur Anpassung der Clusterkapazität. Daher gibt es eine inhärente Latenz im Zusammenhang mit der Veröffentlichung der CloudWatch Metriken, der Zeit, die die Metrik benötigt, `CapacityProviderReservation` um CloudWatch Alarme zu verletzen (sowohl hoch als auch niedrig), und der Zeit, die eine neu gestartete Amazon EC2 EC2-Instance zum Aufwärmen benötigt. Sie können die folgenden Maßnahmen ergreifen, um die auto Clusterskalierung für schnellere Bereitstellungen reaktionsschneller zu gestalten:

Schrittweise Skalierung der Größen durch Kapazitätsanbieter

Amazon ECS-Kapazitätsanbieter werden die Container-Instances irgendwann vergrößern oder verkleinern, um den Anforderungen Ihrer Anwendung gerecht zu werden. Die Mindestanzahl von Instances, die Amazon ECS startet, ist standardmäßig auf 1 festgelegt. Dies kann Ihre Bereitstellungen verlängern, wenn mehrere Instances für die Ausführung Ihrer ausstehenden Aufgaben erforderlich sind. Sie können die `minimumScalingStepSize`über die Amazon ECS-API erhöhen, um die Mindestanzahl von Instances zu erhöhen, die Amazon ECS gleichzeitig ein- oder ausskaliert. Ein zu niedriger Wert kann einschränken `maximumScalingStepSize`, wie viele Container-Instances gleichzeitig ein- oder ausgeschaltet werden, was Ihre Bereitstellungen verlangsamen kann.

> **Note**
>
> Diese Konfiguration ist derzeit nur über die APIs `CreateCapacityProvider`oder `UpdateCapacityProvider`verfügbar.

Aufwärmphase der Instanz

Die Instance-Aufwärmphase ist der Zeitraum, nach dem eine neu gestartete Amazon EC2 EC2-Instance zu den CloudWatch Metriken für die Auto Scaling Scaling-Gruppe beitragen kann.

Sobald die angegebene Aufwärmphase abgelaufen ist, wird die Instance auf die aggregierten Metriken der Auto Scaling-Gruppe angerechnet, und Cluster-Auto Scaling fährt mit der nächsten Berechnungsiteration fort, um die Anzahl der benötigten Instances zu schätzen.

Der Standardwert für `instanceWarmupPeriod`ist 300 Sekunden, den Sie über die `UpdateCapacityProvider`APIs `CreateCapacityProvider`oder auf einen niedrigeren Wert konfigurieren können, um eine reaktionsschnellere Skalierung zu erzielen.

Reservekapazität

Wenn Ihr Kapazitätsanbieter keine Container-Instances für die Platzierung von Aufgaben zur Verfügung hat, muss er die Clusterkapazität erhöhen (skalieren), indem er Amazon EC2 EC2-Instances im laufenden Betrieb startet und wartet, bis sie hochgefahren sind, bevor er Container auf ihnen starten kann. Dies kann die Startrate von Aufgaben erheblich senken. Sie haben hier zwei Möglichkeiten.

In diesem Fall erhöht sich die effektive Startrate von Aufgaben, wenn Amazon EC2 EC2-Kapazitäten bereits gestartet und bereit sind, Aufgaben auszuführen, vorhanden sind. Sie können die `Target Capacity` Konfiguration verwenden, um anzugeben, dass Sie freie Kapazitäten in Ihren Clustern beibehalten möchten. Wenn Sie beispielsweise einen Wert von 80% festlegen`Target Capacity`, geben Sie an, dass Ihr Cluster jederzeit 20% freie Kapazität benötigt. Dank dieser freien Kapazität können alle eigenständigen Aufgaben sofort gestartet werden, wodurch sichergestellt wird, dass das Starten von Aufgaben nicht gedrosselt wird. Der Nachteil dieses Ansatzes sind potenziell höhere Kosten für die Beibehaltung von Clusterkapazitäten.

Ein alternativer Ansatz, den Sie in Betracht ziehen können, besteht darin, Ihrem Service mehr Spielraum zu verleihen, nicht dem Kapazitätsanbieter. Das bedeutet, dass Sie, anstatt die `Target Capacity` Konfiguration zu reduzieren, um freie Kapazitäten bereitzustellen, die Anzahl der Replikate in Ihrem Service erhöhen können, indem Sie die Ziel-Tracking-Skalierungsmetrik oder die Schwellenwerte für die schrittweise Skalierung des Service Auto Scaling ändern. Beachten Sie, dass dieser Ansatz nur bei stark beanspruchten Workloads hilfreich ist, aber keine Auswirkungen hat, wenn Sie neue Services bereitstellen und zum ersten Mal von 0 auf N Aufgaben wechseln. Weitere Informationen zu den zugehörigen Skalierungsrichtlinien finden Sie unter Target Tracking Scaling Policies oder Step Scaling Policies im Amazon Elastic Container Service Developer Guide.

Steuern Sie die Instanzen, die Amazon ECS beendet

> **Important**
>
> Sie müssen den Abskalierungsschutz für Instances von Auto Scaling in der Auto-Scaling-Gruppe aktivieren, um das Feature des verwalteten Beendigungsschutzes der automatischen Cluster-Skalierung zu nutzen.

Mit dem Managed Termination Protection kann Cluster Auto Scaling steuern, welche Instanzen beendet werden. Wenn Sie den Managed Termination Protection verwendet haben, beendet Amazon ECS nur EC2-Instances, für die keine Amazon ECS-Aufgaben ausgeführt werden. Aufgaben, die von einem Service ausgeführt werden, der die `DAEMON`-Planungsstrategie verwendet, werden ignoriert und eine Instance kann durch das Auto Scaling des Clusters beendet werden, auch wenn die Instance diese Aufgaben ausführt. Das liegt daran, dass alle Instances im Cluster diese Aufgaben ausführen.

Amazon ECS aktiviert zunächst die Option Instance Scale-In Protection für die EC2-Instances in der Auto Scaling Scaling-Gruppe. Anschließend platziert Amazon ECS die Aufgaben auf den Instances. Wenn alle Nicht-Daemon-Aufgaben auf einer Instance angehalten werden, initiiert Amazon ECS den Abskalierungs-Prozess und deaktiviert den Abskalierungs-Schutz für die EC2-Instance. Die Auto-Scaling-Gruppe kann dann die Instance beenden.

Der Auto-Scaling-Instance-Abskalierungsschutz steuert, welche EC2-Instances von Auto Scaling beendet werden können. Instances mit aktiviertem Abskalierungs-Feature können während des Abskalierungs-Prozess nicht beendet werden. Weitere Informationen über den Abskalierungsschutz von Auto-Scaling-Instances finden Sie unter Verwenden des Abskalierungsschutzes von Instances im Benutzerhandbuch zu Amazon EC2 Auto Scaling.

Sie können den `targetCapacity` Prozentsatz so festlegen, dass Sie über freie Kapazität verfügen. Dadurch können future Aufgaben schneller gestartet werden, da die Auto Scaling Scaling-Gruppe nicht mehr Instances starten muss. Amazon ECS verwendet den Zielkapazitätswert, um die CloudWatch Metrik zu verwalten, die der Service erstellt. Amazon ECS verwaltet die CloudWatch Metrik. Die Auto Scaling Scaling-Gruppe wird als stationärer Zustand behandelt, sodass keine Skalierungsaktion erforderlich ist. Die Werte können zwischen 0 und 100 % liegen. Um Amazon ECS beispielsweise so zu konfigurieren, dass 10 % freie Kapazität zusätzlich zu der von Amazon-ECS-Aufgaben verwendeten Kapazität beibehalten werden, legen Sie den Zielkapazitätswert

auf 90 % fest. Berücksichtigen Sie Folgendes, wenn Sie den `targetCapacity`-Wert für einen Kapazitätsanbieter festlegen.

- Ein `targetCapacity`-Wert von weniger als 100 % entspricht der Höhe der freien Kapazität (Amazon-EC2-Instaces), die im Cluster vorhanden sein muss. Freie Kapazität bedeutet, dass es keine laufenden Aufgaben gibt.
- Platzierungseinschränkungen wie Availability Zones ohne zusätzliches `binpack` zwingen Amazon ECS dazu, eine Aufgabe für jede Instance auszuführen, was möglicherweise nicht das gewünschte Verhalten ist.

Sie müssen den Auto-Scaling-Instance-Abskalierungsschutz in der Auto-Scaling-Gruppe aktivieren, um den verwalteten Beendigungsschutz zu nutzen. Wenn Sie den Abskalierungsschutz nicht aktivieren, kann die Aktivierung des verwalteten Beendigungsschutzes zu unerwünschtem Verhalten führen. Möglicherweise haben Sie z. B. Instances, die im Ausgleichsstatus feststecken. Weitere Informationen finden Sie unter Instance-Skalierungsschutz verwenden im Benutzerhandbuch zum Amazon EC2 Auto Scaling.

Wenn Sie den Beendigungsschutz mit einem Kapazitätsanbieter verwenden, führen Sie keine manuellen Aktionen, wie z. B. das Trennen der Instance, an der Auto-Scaling-Gruppe durch, die dem Kapazitätsanbieter zugeordnet ist. Manuelle Aktionen können den Abskalierungs-Vorgang des Kapazitätsanbieters unterbrechen. Wenn Sie eine Instance von der Auto-Scaling-Gruppe trennen, müssen Sie die getrennte Instance auch aus dem Amazon-ECS-Cluster abmelden.

Verhalten für die verwaltete Skalierung nach oben

Wenn Sie Auto Scaling Scaling-Gruppenkapazitätsanbieter haben, die verwaltete Skalierung verwenden, schätzt Amazon ECS die optimale Anzahl von Instances, die Sie Ihrem Cluster hinzufügen können, und bestimmt anhand des Werts, wie viele Instances angefordert werden sollen.

Amazon ECS wählt für jede Aufgabe einen Kapazitätsanbieter aus und folgt dabei der Kapazitätsanbieterstrategie des Service, der eigenständigen Aufgabe oder der Cluster-Standardeinstellung. Amazon ECS folgt den restlichen Schritten für einen einzelnen Kapazitätsanbieter.

Aufgaben ohne Kapazitätsanbieterstrategie werden von Kapazitätsanbietern ignoriert. Eine ausstehende Aufgabe, für die es keine Kapazitätsanbieterstrategie gibt, führt bei keinem Kapazitätsanbieter zum Aufskalieren. Aufgaben oder Services können keine Strategie für einen Kapazitätsanbieter festlegen, wenn diese Aufgabe oder dieser Service einen Starttyp festlegt.

Im Folgenden wird das Scale-Out-Verhalten detaillierter beschrieben.

- Gruppieren Sie alle Bereitstellungsaufgaben für diesen Kapazitätsanbieter so, dass jede Gruppe genau die gleichen Ressourcenanforderungen hat.
- Wenn Sie mehrere Instance-Typen in einer Auto-Scaling-Gruppe verwenden, werden die Instance-Typen in der Auto-Scaling-Gruppe nach ihren Parametern sortiert. Zu diesen Parametern gehören vCPU, Arbeitsspeicher, Elastic-Network-Schnittstellen (ENIs), Ports und GPUs. Die kleinsten und größten Instance-Typen für jeden Parameter werden ausgewählt. Weitere Informationen zur Auswahl des Instance-Typs finden Sie unterAmazon EC2 EC2-Container-Instances für Amazon ECS.

  > **Important**
  >
  > Wenn eine Gruppe von Aufgaben einen Ressourcenbedarf hat, der größer ist als der kleinste Instance-Typ in der Auto-Scaling-Gruppe, dann kann diese Gruppe von Aufgaben nicht mit diesem Kapazitätsanbieter ausgeführt werden. Der Kapazitätsanbieter skaliert die Auto-Scaling-Gruppe nicht. Die Aufgaben verbleiben im Status `PROVISIONING`.
  > Um zu verhindern, dass Aufgaben im Status `PROVISIONING` verbleiben, empfehlen wir Ihnen, separate Auto-Scaling-Gruppen und Kapazitätsanbieter für unterschiedliche Mindestressourcenanforderungen zu erstellen. Wenn Sie Aufgaben ausführen oder Services erstellen, fügen Sie nur Kapazitätsanbieter zur Kapazitätsanbieterstrategie hinzu, die die Aufgabe auf dem kleinsten Instance-Typ in der Auto-Scaling-Gruppe ausführen können. Für andere Parameter können Sie Platzierungsbeschränkungen verwenden

- Für jede Aufgabengruppe berechnet Amazon ECS die Anzahl der Instances, die zum Ausführen der nicht platzierten Aufgaben erforderlich sind. Diese Berechnung verwendet eine `binpack`-Strategie. Diese Strategie berücksichtigt die Anforderungen der Aufgaben an vCPU, Arbeitsspeicher, Elastic-Network-Schnittstellen (ENI), Ports und GPUs. Es berücksichtigt auch die Ressourcenverfügbarkeit der Amazon-EC2-Instances. Die Werte für die größten Instance-Typen werden als maximale berechnete Instance-Anzahl behandelt. Die Werte für den kleinsten Instance-Typ werden als Schutz verwendet. Wenn der kleinste Instance-Typ nicht mindestens eine Instance der Aufgabe ausführen kann, betrachtet die Berechnung die Aufgabe als nicht kompatibel. Daher wird die Aufgabe von der Aufskalierungsberechnung ausgeschlossen. Wenn alle Aufgaben nicht mit dem kleinsten Instance-Typ kompatibel sind, stoppt das Auto Scaling des Clusters und der Wert `CapacityProviderReservation` bleibt auf dem Wert `targetCapacity`.
- Amazon ECS veröffentlicht die `CapacityProviderReservation` Metrik in CloudWatch Bezug darauf, `minimumScalingStepSize` ob einer der folgenden Punkte zutrifft.

  - Die maximale berechnete Instance-Anzahl ist geringer als die minimale Skalierungsschrittgröße.
  - Der niedrigere Wert der `maximumScalingStepSize` oder der maximalen berechneten Instanzanzahl.
- CloudWatch Alarme verwenden die `CapacityProviderReservation` Metrik für Kapazitätsanbieter. Wenn die `CapacityProviderReservation`-Metrik größer als der `targetCapacity`-Wert ist, erhöhen Alarme auch die `DesiredCapacity` der Auto-Scaling-Gruppe. Der `targetCapacity` Wert ist eine Einstellung des Kapazitätsanbieters, die während der Aktivierungsphase für die auto Skalierung des Clusters an den CloudWatch Alarm gesendet wird.

  Die Standardeinstellung `targetCapacity` ist 100%.
- Die Auto-Scaling-Gruppe launcht zusätzliche EC2-Instances. Um eine übermäßige Bereitstellung zu verhindern, stellt Auto Scaling sicher, dass die Kapazität der kürzlich gestarteten EC2-Instances stabilisiert wird, bevor neue Instances gestartet werden. Auto Scaling prüft, ob alle vorhandenen Instances die `instanceWarmupPeriod` (jetzt abzüglich der Startzeit der Instance) überschritten haben. Das Scale-Out ist für Instances blockiert, die sich innerhalb von befinden. `instanceWarmupPeriod`

  Die Standardanzahl von Sekunden für das Aufwärmen einer neu gelaunchten Instance beträgt 300.

Weitere Informationen finden Sie unter Detaillierte Informationen zum Auto Scaling von Amazon-ECS-Clustern.

Überlegungen zum Aufskalieren

Berücksichtigen Sie Folgendes für den Aufskalierungs-Prozess:

- Obwohl es mehrere Platzierungs-Einschränkungen gibt, empfehlen wir, dass Sie nur die `distinctInstance`-Einschränkung für die Aufgabenplatzierung nutzen. Dies verhindert, dass der Aufskalierungs-Prozess angehalten wird, da Sie eine Platzierungs-Einschränkung verwenden, die nicht mit den Stichproben-Instances kompatibel ist.
- Die verwaltete Skalierung funktioniert am besten, wenn Ihre Auto-Scaling-Gruppe dieselben oder ähnliche Instance-Typen verwendet.
- Wenn ein Aufskalierungsprozess erforderlich ist und derzeit keine Container-Instances laufen, skaliert Amazon ECS zunächst immer auf zwei Instances und führt dann weitere Aufskalierungs- oder Abskalierungsprozesse durch. Bei jedem weiteren Aufskalieren wird auf die Aufwärmphase der Instance gewartet. Bei Aufskalierungsprozessen wartet Amazon ECS nach einem Aufskalierungsprozess immer 15 Minuten, bevor Abskalierungsprozesse gestartet werden.

- Der zweite Aufskalierungs-Schritt muss warten, bis das `instanceWarmupPeriod` abläuft, was sich auf die Gesamtskalierungsgrenze auswirken kann. Wenn Sie diese Zeit reduzieren müssen, stellen Sie sicher, dass sie groß genug `instanceWarmupPeriod` ist, damit die EC2-Instance den Amazon ECS-Agenten starten und starten kann (wodurch eine übermäßige Bereitstellung verhindert wird).
- Cluster-Auto-Scaling unterstützt Startkonfiguration, Startvorlagen und mehrere Instance-Typen in der Auto-Scaling-Gruppe des Kapazitätsanbieters. Sie können auch die attributbasierte Instance-Typauswahl ohne mehrfache Instance-Typen verwenden.
- Wenn Sie eine Auto-Scaling-Gruppe mit On-Demand-Instances und mehreren Instance-Typen oder Spot-Instances verwenden, platzieren Sie die größeren Instance-Typen höher in der Prioritätsliste und geben Sie keine Gewichtung an. Die Angabe einer Gewichtung wird derzeit nicht unterstützt. Weitere Informationen finden Sie unter Auto-Scaling-Gruppen mit mehreren Instance-Typen im AWS Auto Scaling -Benutzerhandbuch.
- Amazon ECS startet dann entweder `minimumScalingStepSize`, wenn die maximal berechnete Instance-Anzahl kleiner als die minimale Skalierungsschrittgröße ist, oder den niedrigeren Wert von `maximumScalingStepSize` oder der maximal berechneten Instance-Anzahl.
- Wenn ein Amazon ECS-Service oder eine Aufgabe `run-task` startet und die Container-Instances des Kapazitätsanbieters nicht über genügend Ressourcen verfügen, um die Aufgabe zu starten, begrenzt Amazon ECS die Anzahl der Aufgaben mit diesem Status für jeden Cluster und verhindert, dass Aufgaben dieses Limit überschreiten. Weitere Informationen finden Sie unter Servicekontingente.

Verwaltetes Abskalierungs-Verhalten

Amazon ECS überwacht Container-Instances für jeden Kapazitätsanbieter in einem Cluster. Wenn eine Container-Instance keine Aufgaben ausführt, gilt die Container-Instance als leer und Amazon ECS startet den Abskalierungsprozess.

CloudWatch Scale-In-Alarme benötigen 15 Datenpunkte (15 Minuten), bevor der Scale-In-Prozess für die Auto Scaling Scaling-Gruppe beginnt. Nachdem der Abskalierungs-Prozess gestartet wurde, bis Amazon ECS die Anzahl der angemeldeten Container-Instances reduzieren muss, legt die Auto-Scaling-Gruppe fest, dass der `DesireCapacity`-Wert größer als eine Instance und weniger als 50 % pro Minute sein soll.

Wenn Amazon ECS eine Aufskalierung anfordert (wenn `CapacityProviderReservation` größer als 100 ist), während ein Abskalierungs-Prozess durchgeführt wird, wird der Abskalierungs-Prozess angehalten und bei Bedarf von Neuem gestartet.

Im Folgenden wird das Abskalierungsverhalten im Detail beschrieben:

1. Amazon ECS berechnet die Anzahl der leeren Container-Instances. Eine Container-Instance gilt als leer, auch wenn keine Daemon-Aufgaben ausgeführt werden.
2. Amazon ECS setzt den Wert `CapacityProviderReservation` auf eine Zahl zwischen 0–100, die anhand der folgenden Formel das Verhältnis zwischen der erforderlichen Größe der Auto-Scaling-Gruppe und der tatsächlichen Größe darstellt, ausgedrückt in Prozent. Anschließend veröffentlicht Amazon ECS die Metrik für CloudWatch. Weitere Informationen darüber, wie die Metrik berechnet wird, finden Sie unter Deep Dive im Cluster-Auto-Scaling von Amazon-ECS

   ```
   CapacityProviderReservation = (number of instances needed) / (number of running
    instances) x 100
   ```

3. Die `CapacityProviderReservation` Metrik generiert einen CloudWatch Alarm. Dieser Alarm aktualisiert den `DesiredCapacity`-Wert für die Auto-Scaling-Gruppe. Anschließend wird eine der folgenden Aktionen ausgeführt:
   - Wenn Sie keine vom Kapazitätsanbieter verwaltete Beendigung verwenden, wählt die Auto-Scaling-Gruppe EC2-Instances mithilfe der Auto-Scaling-Gruppen-Beendigungsrichtlinie aus und beendet die Instances, bis die Anzahl der EC2-Instances die `DesiredCapacity` erreicht. Die Container-Instances werden dann vom Cluster abgemeldet.
   - Wenn alle Container-Instances einen verwalteten Beendigungsschutz verwenden, entfernt Amazon ECS den Abskalierungsschutz für leere Container-Instances. Die Auto-Scaling-Gruppe kann dann die EC2-Instances beenden. Die Container-Instances werden dann vom Cluster abgemeldet.

auto Skalierung des Amazon ECS-Clusters aktivieren

Sie können das verwenden AWS CLI , um die auto Clusterskalierung zu aktivieren.

Erstellen Sie vor Beginn eine Auto-Scaling-Gruppe und einen Kapazitätsanbieter. Weitere Informationen finden Sie unter the section called “Kapazitätsanbieter für den EC2-Starttyp”.

Um Cluster Auto Scaling zu aktivieren, verknüpfen Sie den Kapazitätsanbieter mit dem Cluster. Anschließend aktivieren Sie Cluster Auto Scaling.

1. Führen Sie den Befehl `put-cluster-capacity-providers` aus, um dem Cluster einen oder mehrere Kapazitätsanbieter zuzuordnen.

Um die AWS Fargate Kapazitätsanbieter hinzuzufügen, schließen Sie die FARGATE und die FARGATE_SPOT Kapazitätsanbieter in die Anfrage ein. Weitere Informationen finden Sie unter put-cluster-capacity-providers in der Referenz zum AWS CLI -Befehl.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers CapacityProviderName FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
```

Um eine Auto Scaling Scaling-Gruppe für den EC2-Starttyp hinzuzufügen, geben Sie den Namen der Auto Scaling Scaling-Gruppe in die Anfrage ein. Weitere Informationen finden Sie unter put-cluster-capacity-providers in der Referenz zum AWS CLI -Befehl.

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers CapacityProviderName \
  --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
```

2. Führen Sie den Befehl describe-clusters aus, um zu überprüfen, ob die Zuordnung erfolgreich war. Weitere Informationen finden Sie unter describe-clusters in der Referenz zum AWS CLI -Befehl.

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

3. Führen Sie den Befehl update-capacity-provider aus, um das verwaltete Auto Scaling für den Kapazitätsanbieter zu aktivieren. Weitere Informationen finden Sie unter update-capacity-provider in der Referenz zum AWS CLI -Befehl.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=ENABLED
```

auto Skalierung des Amazon ECS-Clusters ausschalten

Sie können das verwenden AWS CLI , um die auto Clusterskalierung zu deaktivieren.

Um die auto Clusterskalierung für einen Cluster zu deaktivieren, können Sie entweder den Kapazitätsanbieter mit aktivierter verwalteter Skalierung vom Cluster trennen oder den Kapazitätsanbieter aktualisieren, um die verwaltete Skalierung zu deaktivieren.

Trennen Sie die Zuordnung des Kapazitätsanbieters

Führen Sie die folgenden Schritte aus, um den Kapazitätsanbieter von einem Cluster zu trennen.

1. Verwenden Sie den Befehl `put-cluster-capacity-providers`, um den Kapazitätsanbieter für Auto-Scaling-Gruppen vom Cluster zu trennen. Der Cluster kann die Verbindung zu den AWS Fargate Kapazitätsanbietern beibehalten. Weitere Informationen finden Sie unter `put-cluster-capacity-providers` in der Referenz zum AWS CLI -Befehl.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers FARGATE FARGATE_SPOT \
     --default-capacity-provider-strategy '[]'
   ```

   Verwenden Sie den Befehl `put-cluster-capacity-providers`, um den Kapazitätsanbieter für Auto-Scaling-Gruppen vom Cluster zu trennen. Weitere Informationen finden Sie unter `put-cluster-capacity-providers` in der Referenz zum AWS CLI -Befehl.

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers [] \
     --default-capacity-provider-strategy '[]'
   ```

2. Führen Sie den Befehl `describe-clusters` aus, um zu überprüfen, ob die Trennung erfolgreich war. Weitere Informationen finden Sie unter `describe-clusters` in der Referenz zum AWS CLI -Befehl.

   ```
   aws ecs describe-clusters \
     --cluster ClusterName \
     --include ATTACHMENTS
   ```

Deaktivieren Sie die verwaltete Skalierung für den Kapazitätsanbieter

Führen Sie die folgenden Schritte aus, um die verwaltete Skalierung für den Kapazitätsanbieter zu deaktivieren.

- Führen Sie den Befehl `update-capacity-provider` aus, um das verwaltete Auto Scaling für den Kapazitätsanbieter zu deaktivieren. Weitere Informationen finden Sie unter update-capacity-provider in der Referenz zum AWS CLI -Befehl.

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=DISABLED
```

Beenden Sie sicher Amazon ECS-Workloads, die auf EC2-Instances ausgeführt werden

Managed Instance Draining ermöglicht die ordnungsgemäße Kündigung von Amazon EC2 EC2-Instances. Auf diese Weise können Ihre Workloads sicher gestoppt und auf Instances ohne Terminierung verschoben werden. Wartung und Updates der Infrastruktur werden durchgeführt, ohne dass Sie sich Gedanken über Unterbrechungen der Arbeitslast machen müssen. Durch die Verwendung von Managed Instance Draining vereinfachen Sie Ihre Workflows zur Infrastrukturverwaltung, die den Austausch von Amazon EC2 EC2-Instances erfordern, und sorgen gleichzeitig für Stabilität und Verfügbarkeit Ihrer Anwendungen.

Das Managed Instance Draining von Amazon ECS funktioniert mit dem Ersatz von Auto Scaling Scaling-Gruppeninstanzen. Auf der Grundlage der Instance-Aktualisierung und der maximalen Instance-Lebensdauer können Kunden sicherstellen, dass sie die neuesten Betriebssystem- und Sicherheitsvorschriften für ihre Kapazität einhalten.

Managed Instance Draining kann nur mit Amazon ECS-Kapazitätsanbietern verwendet werden. Sie können Managed Instance Draining aktivieren, wenn Sie Ihre Auto Scaling Scaling-Gruppenkapazitätsanbieter mithilfe der Amazon ECS-Konsole oder des SDK erstellen oder aktualisieren. AWS CLI

Die folgenden Ereignisse werden durch Amazon ECS Managed Instance Draining abgedeckt.

- Auto Scaling Scaling-Gruppen-Instance-Aktualisierung — Verwenden Sie die Instance-Aktualisierung, um Ihre Amazon EC2 EC2-Instances in Ihrer Auto Scaling Scaling-Gruppe fortlaufend zu ersetzen, anstatt dies manuell stapelweise durchzuführen. Dies ist nützlich, wenn Sie eine große Anzahl von Instances ersetzen müssen. Eine Instance-Aktualisierung wird über die Amazon EC2 EC2-Konsole oder die `StartInstanceRefresh` API initiiert. Stellen Sie sicher, dass Sie beim Telefonieren Scale-in-Schutz auswählen`Replace`, `StartInstanceRefresh` wenn Sie den Managed Termination Protection verwenden.

- Maximale Instance-Lebensdauer - Sie können eine maximale Lebensdauer definieren, wenn es darum geht, Auto Scaling Scaling-Gruppeninstanzen zu ersetzen. Dies ist hilfreich für die Planung von Ersatzinstanzen auf der Grundlage interner Sicherheitsrichtlinien oder der Einhaltung von Vorschriften.
- Auto Scaling-Gruppenskalierung - Basierend auf Skalierungsrichtlinien und geplanten Skalierungsaktionen unterstützt die Auto Scaling Scaling-Gruppe die Auto Scaling von Instances. Durch die Verwendung einer Auto Scaling Scaling-Gruppe als Amazon ECS-Kapazitätsanbieter können Sie Auto Scaling-Gruppen-Instances skalieren, wenn in ihnen keine Aufgaben ausgeführt werden.
- Zustandsprüfungen für Auto Scaling Scaling-Gruppen — Die Auto Scaling Scaling-Gruppe unterstützt viele Integritätsprüfungen, um die Kündigung fehlerhafter Instances zu verwalten.
- AWS CloudFormation Stack-Updates - Sie können Ihrem AWS CloudFormation Stack ein `UpdatePolicy` Attribut hinzufügen, um fortlaufende Aktualisierungen durchzuführen, wenn sich die Gruppe ändert.
- Neuausrichtung der Spot-Kapazität — Die Auto Scaling Scaling-Gruppe versucht, Spot-Instances, bei denen ein höheres Ausfallrisiko besteht, proaktiv zu ersetzen, basierend auf der Amazon EC2 EC2-Mitteilung zur Kapazitätsumverteilung. Die Auto Scaling Scaling-Gruppe beendet die alte Instance, wenn die Ersatzinstanz gestartet und fehlerfrei ist. Beim Entleeren einer von Amazon ECS verwalteten Instance wird die Spot-Instance genauso entleert wie eine Nicht-Spot-Instance.
- Spot-Unterbrechung — Spot-Instances werden mit einer Frist von zwei Minuten beendet. Das von Amazon ECS verwaltete Instance-Draining versetzt die Instance daraufhin in den Draining-Status.

Lebenszyklus-Hooks von Amazon EC2 Auto Scaling mit Managed Instance Draining

Auto Scaling Scaling-Gruppenlebenszyklus-Hooks ermöglichen es Kunden, Lösungen zu erstellen, die durch bestimmte Ereignisse im Instanzlebenszyklus ausgelöst werden, und eine benutzerdefinierte Aktion auszuführen, wenn dieses bestimmte Ereignis eintritt. Eine Auto Scaling Scaling-Gruppe ermöglicht bis zu 50 Hooks. Es können mehrere Terminierungs-Hooks existieren, die parallel ausgeführt werden, und die Auto Scaling Scaling-Gruppe wartet, bis alle Hooks abgeschlossen sind, bevor sie eine Instance beenden.

Neben der von Amazon ECS verwalteten Hook-Termination können Sie auch Ihre eigenen Lifecycle-Termination-Hooks konfigurieren. Lifecycle-Hooks haben eine`default action`, und wir empfehlen, diese Einstellung `continue` als Standard festzulegen, um sicherzustellen, dass andere Hooks, wie der von Amazon ECS verwaltete Hook, nicht von Fehlern durch benutzerdefinierte Hooks beeinträchtigt werden.

Wenn Sie bereits einen Lifecycle-Hook zur Beendigung von Auto Scaling Scaling-Gruppen konfiguriert und auch das Managed Instance Draining von Amazon ECS aktiviert haben, werden beide Lifecycle-Hooks ausgeführt. Die jeweiligen Zeitpunkte können jedoch nicht garantiert werden. Lifecycle-Hooks haben eine `default action` Einstellung, mit der festgelegt wird, welche Aktion ausgeführt werden soll, wenn das Timeout abgelaufen ist. Im Falle von Fehlern empfehlen wir, dies `continue` als Standardergebnis in Ihrem benutzerdefinierten Hook zu verwenden. Dadurch wird sichergestellt, dass andere Hooks, insbesondere die von Amazon ECS verwalteten Hooks, nicht durch Fehler in Ihrem benutzerdefinierten Lifecycle-Hook beeinträchtigt werden. Das alternative Ergebnis von `abandon` führt dazu, dass alle anderen Hooks übersprungen werden und sollten vermieden werden. Weitere Informationen zu Auto Scaling Scaling-Gruppen-Lifecycle-Hooks finden Sie unter Amazon EC2 Auto Scaling Scaling-Lifecycle-Hooks im Amazon EC2 Auto Scaling Scaling-Benutzerhandbuch.

Aufgaben und Managed Instance Draining

Das von Amazon ECS verwaltete Instance-Draining verwendet die bestehende Draining-Funktion, die in Container-Instances zu finden ist. Die Funktion zum Entleeren von Container-Instances führt den Austausch durch und stoppt bei Replikataufgaben, die zu einem Amazon ECS-Service gehören. Eine eigenständige Aufgabe, z. B. eine von aufgerufene`RunTask`, die sich im `RUNNING` Status `PENDING` oder befindet, bleibt davon unberührt. Sie müssen warten, bis diese entweder abgeschlossen oder manuell beendet sind. Die Container-Instance bleibt so lange im `DRAINING` Status, bis entweder alle Aufgaben gestoppt wurden oder 48 Stunden vergangen sind. Daemon-Aufgaben werden als letzte beendet, nachdem alle Replikataufgaben beendet wurden.

Managed Instance Draining und verwalteter Kündigungsschutz

Managed Instance Draining funktioniert auch dann, wenn die verwaltete Kündigung deaktiviert ist. Informationen zum verwalteten Kündigungsschutz finden Sie unterSteuern Sie die Instanzen, die Amazon ECS beendet.

In der folgenden Tabelle wird das Verhalten verschiedener Kombinationen aus verwalteter Kündigung und verwaltetem Leerlauf zusammengefasst.

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
Aktiviert	Aktiviert	Amazon ECS schützt Amazon

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
		EC2 EC2-Instances, auf denen Aufgaben ausgeführt werden, davor, durch Scale-In-Ereignisse beendet zu werden. Alle Instances, die beendet werden, z. B. solche, für die kein Kündigungsschutz aktiviert ist, bei denen Spot-Unterbrechungen aufgetreten sind oder die durch eine Instance-Aktualisierung erzwungen wurden,

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
		werden ordnungsgemäß gelöscht.
Disabled	Aktiviert	Amazon ECS schützt Amazon EC2 EC2-Instances, auf denen Aufgaben ausgeführt werden, nicht davor, skaliert zu werden. Alle Instances, die beendet werden, werden jedoch ordnungsgemäß gelöscht.

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
Enabled	Disabled	Amazon ECS schützt Amazon EC2 EC2-Instances, auf denen Aufgaben ausgeführt werden, davor, durch Scale-In-Ereignisse beendet zu werden. Instances können jedoch immer noch durch Spot-Unterbrechungen oder erzwungene Instance-Aktualisierungen beendet werden oder wenn sie keine Aufgaben ausführen

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
		. Amazon ECS führt für diese Instances kein ordnungsgemäßes Entleeren durch und startet Ersatzservice-Aufgaben, nachdem sie beendet wurden.

Verwaltete Kündigung	Verwaltete Entleerung	Ergebnis
Disabled	Disabled	Amazon EC2 EC2-Instances können jederzeit skaliert oder beendet werden, auch wenn sie Amazon ECS-Aufgaben ausführen. Amazon ECS startet Ersatzservice-Aufgaben, nachdem sie beendet wurden.

Entleerung verwalteter Instances und Entleerung von Spot-Instances

Mit Spot-Instance-Draining können Sie eine Umgebungsvariable `ECS_ENABLE_SPOT_INSTANCE_DRAINING` auf dem Amazon ECS-Agenten einrichten, die es Amazon ECS ermöglicht, eine Instance als Reaktion auf die zweiminütige Spot-Unterbrechung in den Status Draining zu versetzen. Das Managed Instance Draining von Amazon ECS ermöglicht das reibungslose Herunterfahren von Amazon EC2 EC2-Instances, die aus vielen Gründen beendet werden, nicht nur wegen Spot-Unterbrechungen. Sie können beispielsweise Amazon EC2 Auto Scaling Capacity Rebalancing verwenden, um Spot-Instances mit erhöhtem

Unterbrechungsrisiko proaktiv zu ersetzen, und das Managed Instance Draining führt dazu, dass die zu ersetzende Spot-Instance ordnungsgemäß heruntergefahren wird. Wenn Sie Managed Instance Draining verwenden, müssen Sie Spot-Instance Draining nicht separat aktivieren, sodass `ECS_ENABLE_SPOT_INSTANCE_DRAINING` in Auto Scaling Gruppenbenutzerdaten redundant sind. Weitere Informationen zum Entladen von Spot-Instances finden Sie unter. Spot-Instances

So funktioniert das Managed Instance Draining mit EventBridge

Von Amazon ECS verwaltete Instance-Draining-Ereignisse werden auf Amazon veröffentlicht EventBridge, und Amazon ECS erstellt eine EventBridge verwaltete Regel im Standardbus Ihres Kontos, um das Managed Instance Draining zu unterstützen. Sie können diese Ereignisse nach anderen AWS Diensten wie Lambda, Amazon SNS und Amazon SQS filtern, um sie zu überwachen und Fehler zu beheben.

- Amazon EC2 Auto Scaling sendet ein Ereignis, EventBridge wenn ein Lifecycle-Hook aufgerufen wird.
- Hinweise zu punktuellen Unterbrechungen werden unter veröffentlicht. EventBridge
- Amazon ECS generiert Fehlermeldungen, die Sie über die Amazon ECS-Konsole und die APIs abrufen können.
- EventBridge verfügt über integrierte Wiederholungsmechanismen, um vorübergehende Ausfälle zu vermeiden.

Konfiguration von Amazon ECS-Kapazitätsanbietern zum sicheren Herunterfahren von Instances

Sie können Managed Instance Draining aktivieren, wenn Sie Ihre Auto Scaling Scaling-Gruppenkapazitätsanbieter mithilfe der Amazon ECS-Konsole und AWS CLI aktualisieren.

> **Note**
>
> Managed Instance Draining ist standardmäßig aktiviert, wenn Sie einen Kapazitätsanbieter erstellen.

Im Folgenden finden Sie Beispiele AWS CLI für die Verwendung von zum Erstellen eines Kapazitätsanbieters mit aktiviertem Managed Instance Draining und zum Aktivieren des Managed Instance Draining für den vorhandenen Kapazitätsanbieter eines Clusters.

Erstellen Sie einen Kapazitätsanbieter mit aktiviertem Managed Instance Draining

Verwenden Sie den Befehl, um einen Kapazitätsanbieter mit aktiviertem Managed Instance Draining zu erstellen. `create-capacity-provider` Stellen Sie den Parameter `managedDraining` auf `ENABLED` ein.

```
aws ecs create-capacity-provider \
--name capacity-provider \
--auto-scaling-group-provider '{
  "autoScalingGroupArn": "asg-arn",
  "managedScaling": {
    "status": "ENABLED",
    "targetCapacity": 100,
    "minimumScalingStepSize": 1,
    "maximumScalingStepSize": 1
  },
  "managedDraining": "ENABLED",
  "managedTerminationProtection": "ENABLED",
}'
```

Antwort:

```
{
    "capacityProvider": {
        "capacityProviderArn": "capacity-provider-arn",
        "name": "capacity-provider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1
            },
            "managedTerminationProtection": "ENABLED"
            "managedDraining": "ENABLED"
        }
    }
}
```

Aktivieren Sie das Managed Instance Draining für den vorhandenen Kapazitätsanbieter eines Clusters

Die Option Managed Instance Draining für den vorhandenen Kapazitätsanbieter eines Clusters aktivieren verwendet den `update-capacity-provider` Befehl. Sie sehen, dass das `managedDraining` derzeit sagt `DISABLED` und `updateStatus` sagt`UPDATE_IN_PROGRESS`.

```
aws ecs update-capacity-provider \
--name cp-draining \
--auto-scaling-group-provider '{
  "managedDraining": "ENABLED"
}
```

Antwort:

```
{
    "capacityProvider": {
        "capacityProviderArn": "cp-draining-arn",
        "name": "cp-draining",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-draining-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1,
                "instanceWarmupPeriod": 300
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "DISABLED" // before update
        },
        "updateStatus": "UPDATE_IN_PROGRESS", // in progress and need describe again to
 find out the result
        "tags": [
        ]
    }
}
```

Benutze den `describe-clusters` Befehl und füge hinzu`ATTACHMENTS`. Der Anhang `status` der verwalteten Instanz ist`PRECREATED`, und der gesamte `attachmentsStatus` ist`UPDATING`.

```
aws ecs describe-clusters --clusters cluster-name --include ATTACHMENTS
```

Antwort:

```
{
    "clusters": [
        {
            ...

            "capacityProviders": [
                "cp-draining"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                # new precreated managed draining attachment
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "type": "managed_draining",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "cp-draining"
                        },
                        {
                            "name": "autoScalingLifecycleHookName",
                            "value": "ecs-managed-draining-termination-hook"
                        }
                    ]
                },

                ...

            ],
            "attachmentsStatus": "UPDATING"
        }
    ],
    "failures": []
}
```

Wenn das Update abgeschlossen ist`describe-capacity-providers`, verwenden Sie es und Sie sehen `managedDraining` es jetzt`ENABLED`.

```
aws ecs describe-capacity-providers --capacity-providers cp-draining
```

Antwort:

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "cp-draining-arn",
            "name": "cp-draining",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "asg-draning-arn",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1,
                    "instanceWarmupPeriod": 300
                },
                "managedTerminationProtection": "DISABLED",
                "managedDraining": "ENABLED" // successfully update
            },
            "updateStatus": "UPDATE_COMPLETE",
            "tags": []
        }
    ]
}
```

Fehlerbehebung beim Entladen von Amazon ECS Managed Instances

Möglicherweise müssen Sie Probleme mit dem Managed Instance Draining beheben. Im Folgenden finden Sie ein Beispiel für ein Problem und eine Lösung, auf die Sie bei der Verwendung stoßen können.

Bei Verwendung von Auto Scaling werden Instances nicht beendet, wenn sie die maximale Instanzlebensdauer überschritten haben.

Wenn Ihre Instances auch nach Erreichen und Überschreitung der maximalen Instance-Lebensdauer bei Verwendung einer Auto Scaling-Gruppe nicht beendet werden, liegt das möglicherweise daran, dass sie vor Scale-In geschützt sind. Sie können die verwaltete Kündigung deaktivieren und Managed Draining zulassen, um das Instance-Recycling zu übernehmen.

Erstellen von Ressourcen für Amazon ECS-Cluster Auto Scaling mit dem AWS Management Console

Erfahren Sie, wie Sie die Ressourcen für die auto Clusterskalierung mithilfe des erstellen AWS Management Console. Wenn Ressourcen einen Namen benötigen, verwenden wir das Präfix, `ConsoleTutorial` um sicherzustellen, dass sie alle eindeutige Namen haben und damit sie leicht auffindbar sind.

Themen

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Die Amazon-ECS-Container-Instance IAM-Rolle wird erstellt. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.
- Die Amazon-ECS-serviceverknüpfte IAM-Rolle wird erstellt. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.
- Die serviceverknüpften IAM-Rolle für Auto Scaling wird erstellt. Weitere Informationen finden Sie unter Servicebezogene Rollen für Amazon EC2 Auto Scaling im Benutzerhandbuch zum Amazon EC2 Auto Scaling.
- Sie haben eine VPC und die zu verwendende Sicherheitsgruppe erstellt. Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.

Schritt 1: Erstellen eines Amazon-ECS-Clusters

Führen Sie die folgenden Schritte aus, um einen Amazon-ECS-Cluster zu erstellen.

Amazon ECS erstellt in Ihrem Namen eine Amazon EC2 Auto Scaling Scaling-Startvorlage und eine Auto Scaling Scaling-Gruppe als Teil des AWS CloudFormation Stacks.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Cluster und dann Cluster erstellen.
3. Geben Sie unter Cluster-Konfiguration für Cluster-Name `ConsoleTutorial-cluster` ein.
4. Löschen Sie unter Infrastruktur die Option AWS Fargate (serverlos) und wählen Sie dann Amazon EC2 EC2-Instances aus. Konfigurieren Sie als Nächstes die Auto-Scaling-Gruppe, die als Kapazitätsanbieter fungiert.
   - Unter der Auto-Scaling-Gruppe (ASG). Wählen Sie Neue ASG erstellen aus, und geben Sie dann die folgenden Details zur Gruppe ein:
     - Wählen Sie für Betriebssystem/Architektur die Option Amazon Linux 2 aus.
     - Wählen Sie unter EC2-Instance-Typ t3.nano.
     - Geben Sie für Capacity (Kapazität) die minimale Anzahl und die maximale Anzahl von Instances ein, die in der Auto-Scaling-Gruppe gelauncht werden sollen.
5. (Optional) Um die Cluster-Tags zu verwalten, erweitern Sie Tags und führen Sie dann eine der folgenden Vorgänge aus:

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

   [Markierung entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert der Markierung.
6. Wählen Sie Erstellen.

Schritt 2: Registrieren einer Aufgabendefinition

Bevor Sie auf Ihrem Cluster eine Aufgabe ausführen können, müssen Sie eine Aufgabendefinition registrieren. Aufgabendefinitionen sind Listen zusammengefasster Container. Im folgenden Beispiel sehen Sie eine einfache Aufgabendefinition, die ein `amazonlinux`-Image aus dem Docker-Hub verwendet und sich einfach im Ruhezustand befindet. Weitere Informationen zu den verfügbaren Parametern für die Aufgabendefinition finden Sie im Abschnitt Amazon-ECS-Aufgabendefinitionen.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
4. Fügen Sie im Textfeld JSON-Editor den folgenden Inhalt ein.

```
{
    "family": "ConsoleTutorial-taskdef",
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "amazonlinux:2",
            "memory": 20,
            "essential": true,
            "command": [
                "sh",
                "-c",
                "sleep infinity"
            ]
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ]
}
```

5. Wählen Sie Erstellen.

Schritt 3: Ausführen einer Aufgabe

Nachdem Sie eine Aufgabendefinition für Ihr Konto registriert haben, können Sie eine Aufgabe im Cluster ausführen. Für dieses Tutorial führen Sie fünf Instances der `ConsoleTutorial-taskdef`-Aufgabendefinition in Ihrem `ConsoleTutorial-cluster`-Cluster aus.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Cluster-Seite -cluster aus. ConsoleTutorial
3. Wählen Sie unter Aufgaben die Option Neue Aufgabe ausführen.
4. Wählen Sie im Abschnitt Umgebung unter Rechenoptionen die Option Kapazitätsanbieter-Strategie aus.
5. Wählen Sie unter Bereitstellungskonfiguration für Anwendungstyp die Option Aufgabe aus.
6. Wählen Sie ConsoleTutorial-taskdef aus der Dropdownliste Family aus.
7. Geben Sie unter Gewünschte Aufgaben den Wert 5 ein.
8. Wählen Sie Erstellen.

Schritt 4: Überprüfen

Zu diesem Zeitpunkt im Tutorial sollten Sie einen Cluster mit fünf aktiven Aufgaben und eine Auto-Scaling-Gruppe mit einem Kapazitätsanbieter haben. Für den Kapazitätsanbieter ist die Amazon-ECS-verwaltete Skalierung aktiviert.

Wir können überprüfen, ob alles ordnungsgemäß funktioniert, indem wir uns die CloudWatch Metriken, die Auto Scaling Scaling-Gruppeneinstellungen und schließlich die Anzahl der Amazon ECS-Cluster-Aufgaben ansehen.

Um die CloudWatch Metriken für Ihren Cluster anzuzeigen

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie auf der Navigationsleiste oben auf dem Bildschirm die Region aus.
3. Wählen Sie im Navigationsbereich unter Metriken Alle Metriken aus.
4. Wählen Sie auf der Seite Alle Metriken unter der Registerkarte Durchsuchen die Option `AWS/ECS/ManagedScaling`.
5. Wählen Sie CapacityProviderName, ClusterName.
6. Aktivieren Sie das Kontrollkästchen, das dem entspricht `ConsoleTutorial-cluster` ClusterName.

7. Ändern Sie auf der Registerkarte Grafisch dargestellte Metriken den Zeitraum auf 30 Sekunden und Statistik auf Maximum.

   Der im Diagramm angezeigte Wert stellt den Zielkapazitätswert für den Kapazitätsanbieter dar. Er sollte bei `100` beginnen, was der zuvor festgelegte Zielkapazitätsprozentsatz ist. Sie sollten eine Skalierung nach oben bis `200` sehen, wodurch ein Alarm für die Zielverfolgungsskalierungsrichtlinie ausgelöst wird. Der Alarm löst dann eine Skalierung der Auto-Scaling-Gruppe nach oben aus.

Gehen Sie wie folgt vor, um Ihre Auto-Scaling-Gruppendetails anzuzeigen, um zu bestätigen, dass die Aktion zur Skalierung nach oben eingetreten ist.

So überprüfen Sie die Skalierung der Auto-Scaling-Gruppe

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Wählen Sie auf der Navigationsleiste oben auf dem Bildschirm die Region aus.
3. Wählen Sie im Navigationsbereich unter Auto Scaling Auto Scaling Groups (Auto Scaling-Gruppe) aus.
4. Wählen Sie die in diesem Tutorial erstellte `ConsoleTutorial-cluster`-Auto-Scaling-Gruppe aus. Sehen Sie sich den Wert unter Gewünschte Kapazität und die Instances auf der Registerkarte Instance-Verwaltung an, um zu bestätigen, dass Ihre Gruppe auf zwei Instances skaliert wurde.

Führen Sie die folgenden Schritte aus, um Ihren Amazon-ECS-Cluster anzuzeigen und um so zu bestätigen, dass die Amazon-EC2-Instances beim Cluster registriert wurden und Ihre Aufgaben in einen `RUNNING`-Status übergehen.

So überprüfen Sie die Instances in der Auto-Scaling-Gruppe

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den `ConsoleTutorial-cluster`-Cluster aus.
4. Bestätigen Sie auf der Registerkarte Aufgaben, dass fünf Aufgaben im Status `RUNNING` angezeigt werden.

Schritt 5: Bereinigen

Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie die damit verknüpften Ressourcen bereinigen, um zu vermeiden, dass für nicht verwendete Ressourcen Kosten entstehen. Das Löschen von Kapazitätsanbietern und Aufgabendefinitionen wird nicht unterstützt, aber mit diesen Ressourcen sind keine Kosten verbunden.

So bereinigen Sie die Tutorial-Ressourcen

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Seite Cluster die Option ConsoleTutorial-cluster aus.
4. Wählen Sie auf der Seite ConsoleTutorial-cluster die Registerkarte Tasks und dann Stop, Stop All aus.
5. Klicken Sie im Navigationsbereich auf Cluster.
6. Wählen Sie auf der Seite Clusters die Option ConsoleTutorial-cluster aus.
7. Wählen Sie oben rechts auf der Seite die Option Cluster löschen aus.
8. Geben Sie im Bestätigungsfeld delete ConsoleTutorial-cluster ein und wählen Sie Löschen aus.
9. Löschen Sie die Auto-Scaling-Gruppen, indem Sie folgende Schritte ausführen.
    a. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
    b. Wählen Sie auf der Navigationsleiste oben auf dem Bildschirm die Region aus.
    c. Wählen Sie im Navigationsbereich unter Auto Scaling Auto Scaling Groups (Auto Scaling-Gruppe) aus.
    d. Wählen Sie die `ConsoleTutorial-cluster`-Auto-Scaling-Gruppe und dann Aktionen aus.
    e. Wählen Sie im Menü Actions (Aktionen) die Option Delete (Löschen) aus. Geben Sie im Bestätigungsfeld Löschen ein und wählen Sie dann Löschen.

Amazon EC2 EC2-Container-Instances für Amazon ECS

Eine Amazon ECS-Container-Instance ist eine Amazon EC2 EC2-Instance, die den Amazon ECS-Container-Agenten ausführt und in einem Cluster registriert ist. Wenn Sie Aufgaben mit Amazon ECS unter Verwendung des EC2-Starttyps, des externen Starttyps oder eines Kapazitätsanbieters

der Auto-Scaling-Gruppe ausführen, werden Ihre Aufgaben auf Ihren aktiven Container-Instances platziert. Sie sind für die Verwaltung und Wartung von Container-Instances verantwortlich.

Sie können zwar Ihr eigenes Amazon EC2 EC2-Instance-AMI erstellen, das die grundlegenden Spezifikationen erfüllt, die für die Ausführung Ihrer containerisierten Workloads auf Amazon ECS erforderlich sind, aber die für Amazon ECS optimierten AMIs sind vorkonfiguriert und wurden von Technikern auf Amazon ECS getestet. AWS Dies ist die einfachste Methode für den Einstieg, mit dem Ihre Container in AWS schnell einsatzbereit werden.

Wenn Sie mit der Konsole einen Cluster erstellen, erstellt Amazon ECS eine Startvorlage für Ihre Instances mit dem neuesten AMI, das dem ausgewählten Betriebssystem zugeordnet ist.

Wenn Sie AWS CloudFormation einen Cluster erstellen, ist der SSM-Parameter Teil der Amazon EC2 EC2-Startvorlage für die Auto Scaling Scaling-Gruppeninstanzen. Sie können die Vorlage so konfigurieren, dass ein dynamischer Systems Manager Manager-Parameter verwendet wird, um zu bestimmen, welches Amazon ECS-optimierte AMI bereitgestellt werden soll. Dieser Parameter stellt sicher, dass der Stack bei jeder Bereitstellung überprüft, ob ein Update verfügbar ist, das auf die EC2-Instances angewendet werden muss. Ein Beispiel für die Verwendung des Systems Manager Manager-Parameters finden Sie unter Erstellen eines Amazon ECS-Clusters mit dem Amazon ECS-optimierten Amazon Linux 2023 AMI im AWS CloudFormation Benutzerhandbuch.

- Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten
- Abrufen von Amazon Bottlerocket ECS-optimierten AMI-Metadaten
- Amazon ECS-optimierte Windows AMI-Metadaten abrufen

Sie können aus den Instance-Typen wählen, die mit Ihrer Anwendung kompatibel sind. Bei größeren Instances können Sie mehr Aufgaben gleichzeitig starten. Bei kleineren Instances können Sie detaillierter skalieren, um Kosten zu sparen. Sie müssen sich nicht für einen einzigen Amazon-EC2-Instance-Typ entscheiden, der für alle Anwendungen in Ihrem Cluster geeignet ist. Stattdessen können Sie mehrere Auto Scaling Scaling-Gruppen erstellen, wobei jede Gruppe einen anderen Instance-Typ hat. Anschließend können Sie für jede dieser Gruppen einen Amazon EC2 EC2-Kapazitätsanbieter erstellen.

Verwenden Sie die folgenden Richtlinien, um die zu verwendenden Instance-Familientypen und den Instance-Typ zu bestimmen:

- Vermeiden Sie die Instance-Typen oder Instance-Familien, die den spezifischen Anforderungen Ihrer Anwendung nicht entsprechen. Wenn Ihre Anwendung beispielsweise eine GPU benötigt, können Sie alle Instance-Typen ausschließen, die keine GPU haben.
- Berücksichtigen Sie Anforderungen wie Netzwerkdurchsatz und Speicher.
- Denken Sie an die CPU und den Arbeitsspeicher. Als allgemeine Regel gilt, dass die CPU und der Arbeitsspeicher groß genug sein müssen, um mindestens ein Replikat der Aufgabe aufzunehmen, die Sie ausführen möchten.

Spot-Instances

Spot-Kapazität kann im Vergleich zu On-Demand-Instances zu erheblichen Kosteneinsparungen führen. Spot-Kapazität ist überschüssige Kapazität, deren Preis deutlich unter dem Preis für On-Demand-Kapazität oder reservierter Kapazität liegt. Spot-Kapazität eignet sich für Batch-Verarbeitungs- und Machine-Learning-Workloads sowie für Entwicklungs- und Staging-Umgebungen. Generell ist sie für alle Workloads geeignet, die vorübergehende Ausfallzeiten tolerieren.

Seien Sie sich über die folgenden Konsequenzen im Klaren, denn die Spot-Kapazität ist möglicherweise nicht immer verfügbar.

- In Zeiten extrem hoher Nachfrage ist die Spot-Kapazität möglicherweise nicht verfügbar. Dies kann dazu führen, dass der Start von Amazon-EC2-Spot-Instances verzögert wird. In diesen Fällen versuchen Amazon-ECS-Services erneut, Aufgaben zu starten, und Gruppen von Amazon-EC2-Auto-Scaling versuchen ebenfalls erneut, Instances zu starten, bis die erforderliche Kapazität verfügbar ist. Amazon EC2 ersetzt Spot-Kapazität nicht durch On-Demand-Kapazität.
- Wenn der Gesamtbedarf an Kapazität steigt, können Spot-Instances und Aufgaben mit einer Warnung von nur zwei Minuten beendet werden. Nach dem Senden der Warnung sollten die Aufgaben gegebenenfalls korrekt heruntergefahren werden, bevor die Instance vollständig beendet wird. Dies trägt dazu bei, die Wahrscheinlichkeit von Fehlern zu minimieren. Weitere Informationen zu einem korrekten Herunterfahren finden Sie unter Korrektes Herunterfahren mit ECS.

Beachten Sie die folgenden Empfehlungen, um Kapazitätsengpässe vor Ort zu minimieren:

- Verwenden Sie mehrere Regionen und Availability Zones – Die Spot-Kapazität variiert je nach Region und Availability Zone. Sie können die Spot-Verfügbarkeit verbessern, indem Sie Ihre Workloads in mehreren Regionen und Availability Zones ausführen. Geben Sie nach Möglichkeit Subnetze in allen Availability Zones in den Regionen an, in denen Sie Ihre Tasks und Instances ausführen.

- Verwenden von mehreren Amazon-EC2-Instance-Typen – Wenn Sie Mixed Instance Policies mit Amazon EC2 Auto Scaling verwenden, werden mehrere Instance-Typen in Ihrer Auto-Scaling-Gruppe gestartet. Dadurch wird sichergestellt, dass eine Anfrage nach Spot-Kapazität bei Bedarf erfüllt werden kann. Um die Zuverlässigkeit zu maximieren und die Komplexität zu minimieren, sollten Sie in Ihrer Mixed Instances Policy Instance-Typen mit ungefähr der gleichen Menge an CPU und Arbeitsspeicher verwenden. Diese Instances können aus einer anderen Generation oder aus Varianten desselben Basis-Instance-Typs stammen. Beachten Sie, dass sie möglicherweise zusätzliche Features enthalten, die Sie möglicherweise nicht benötigen. Ein Beispiel für eine solche Liste könnte m4.large, m5.large, m5a.large, m5d.large, m5n.large, m5dn.large und m5ad.large enthalten. Weitere Informationen finden Sie unter Auto-Scaling-Gruppen mit mehreren Instance-Typen und Kaufoptionen im Amazon EC2 Auto Scaling-Benutzerhandbuch.
- Verwenden Sie die kapazitätsoptimierte Spot-Zuweisungsstrategie – Mit Amazon EC2 Spot können Sie zwischen kapazitäts- und kostenoptimierten Zuweisungsstrategien wählen. Wenn Sie beim Start einer neuen Instance die kapazitätsoptimierte Strategie wählen, wählt Amazon EC2 Spot den Instance-Typ mit der größten Verfügbarkeit in der ausgewählten Availability Zone aus. Dies trägt dazu bei, die Wahrscheinlichkeit zu verringern, dass die Instance kurz nach ihrem Start beendet wird.

Informationen zur Konfiguration von Spot-Kündigungshinweisen auf Ihren Container-Instances finden Sie unter:

- Konfiguration von Amazon ECS-Linux-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen
- Konfiguration von Amazon ECS-Windows-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen

Amazon ECS-optimierte Linux-AMIs

Amazon ECS stellt die für Amazon ECS optimierten AMIs bereit, die mit den Anforderungen und Empfehlungen für die Ausführung Ihrer Container-Workloads vorkonfiguriert sind. Wir empfehlen Ihnen, das Amazon-ECS-optimierte AMI für Amazon Linux 2023 für Ihre Amazon-EC2-Instances zu verwenden, es sei denn, Ihre Anwendung erfordert Instances auf Basis von Amazon EC2 GPU, ein bestimmtes Betriebssystem oder eine Docker-Version, die in diesem AMI noch nicht verfügbar ist. Informationen zu den Amazon Linux 2- und Amazon Linux 2023-Instances finden Sie unter Vergleich von Amazon Linux 2 und Amazon Linux 2023 im Amazon Linux 2023-Benutzerhandbuch. Wenn Sie Ihre Container-Instances von dem neuesten für Amazon ECS optimierten AMI starten, ist

sichergestellt, dass Sie die aktuellen Sicherheitsupdates und die Version des Container-Agenten erhalten. Weitere Informationen, wie eine Instance gestartet wird, finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

Wenn Sie mit der Konsole einen Cluster erstellen, erstellt Amazon ECS eine Startvorlage für Ihre Instances mit dem neuesten AMI, das dem ausgewählten Betriebssystem zugeordnet ist.

Wenn Sie AWS CloudFormation einen Cluster erstellen, ist der SSM-Parameter Teil der Amazon EC2 EC2-Startvorlage für die Auto Scaling Scaling-Gruppeninstanzen. Sie können die Vorlage so konfigurieren, dass ein dynamischer Systems Manager Manager-Parameter verwendet wird, um zu bestimmen, welches Amazon ECS-optimierte AMI bereitgestellt werden soll. Dieser Parameter stellt sicher, dass der Stack bei jeder Bereitstellung überprüft, ob ein Update verfügbar ist, das auf die EC2-Instances angewendet werden muss. Ein Beispiel für die Verwendung des Systems Manager Manager-Parameters finden Sie unter Erstellen eines Amazon ECS-Clusters mit dem Amazon ECS-optimierten Amazon Linux 2023 AMI im AWS CloudFormation Benutzerhandbuch.

Wenn Sie das Amazon ECS-optimierte AMI anpassen müssen, finden Sie weitere Informationen unter Amazon ECS Optimized AMI Build Recipies auf. GitHub

Die Linux-Varianten des Amazon-ECS-optimierten AMI verwenden das Amazon-Linux-2-AMI als Basis. Die AMI-Versionshinweise von Amazon Linux 2 sind ebenfalls verfügbar. Weitere Informationen finden Sie in den Versionshinweisen zu Amazon Linux 2.

Wir empfehlen, ein AMI mit Linux-Kernel 5.10 zu verwenden, da der Linux-Kernel 4.14 end-of-life am 10. Januar 2024 erreicht wurde.

Die folgenden Varianten des Amazon-ECS-optimierten AMI sind für Ihre Amazon-EC2-Instances verfügbar.

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
Amazon Linux 2023	Amazon-ECS-optimiertes AMI für Amazon Linux 2023	Amazon Linux 2023 ist die nächste Generation von Amazon Linux von AWS. In den meisten Fällen empfohlen, um Ihre Amazon-EC2-Instances für	Standardmäßig wird das Amazon-ECS-optimierte AMI für Amazon Linux 2023 mit einem einzelnen 30-GiB-Root-Volume ausgeliefert. Sie können die

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
		Ihre Amazon-EC S-Workloads zu launchen. Weitere Informationen finden Sie unter Was ist Amazon Linux 2023? im Benutzerhandbuch für Amazon Linux 2023.	Größe des 30-GiB-Root-Volumes zum Startzeitpunkt ändern, um den verfügbaren Speicherplatz auf Ihrer Container-Instance zu erhöhen. Dieser Speicher wird für das Betriebssystem sowie für Docker-Images und Metadaten verwendet. Das Standard-Dateisystem für das Amazon-EC S-optimierte AMI für Amazon Linux 2023 ist `xfs` und Docker verwendet den `overlay2`-Speichertreiber. Weitere Informationen finden Sie unter Verwenden des OverlayFS-Speichertreibers in der Docker-Dokumentation.
Amazon Linux 2023 (arm64)	Amazon-ECS-optimiertes AMI für Amazon Linux 2023 (arm64)	Basierend auf Amazon Linux 2023 wird dieses AMI zur Verwendung beim Starten Ihrer Amazon-EC2-Instances empfohlen, die von Arm-basierten AWS -Graviton/Graviton-2-Prozessoren für Ihre Amazon-ECS-Workloads betrieben werden. Weitere Informationen finden Sie unter General Purpose Instances im Amazon EC2 EC2-Benutzerhandbuch. Das Amazon ECS-optimierte Amazon Linux 2023 (arm64) AMI ist nicht vorinstalliert. AWS CLI	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
Amazon Linux 2023 (Neuron)	Amazon Linux 2023 (Neuron)	Basierend auf Amazon Linux 2023 ist dieses AMI für Amazon EC2 Inf1-, Trn1- oder Inf2-Instances. Es ist mit den Treibern AWS Inferentia und AWS Trainium sowie der AWS Neuron-Runtime für Docker vorkonfiguriert, wodurch die Ausführung von Inferenz-Workloads für maschinelles Lernen auf Amazon ECS vereinfacht wird. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für AWS Neuron-Workloads für maschinelles Lernen. Das Amazon ECS-optimierte Amazon Linux 2023 (Neuron) AMI ist nicht vorinstalliert. AWS CLI	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
Amazon Linux 2	Amazon-ECS-optimiertes Amazon-Linux-2-AMI mit Kernel 5.10	Dieses AMI basiert auf Amazon Linux 2 und ist für den Start Ihrer Amazon-EC2-Instances bestimmt, bei denen Sie Linux-Kernel 5.10 anstelle von Kernel 4.14 für Ihre Amazon-ECS-Workloads verwenden sollten. Das Amazon-ECS-optimierte Amazon-Linux-2-AMI Kernel 5.10 ist nicht mit dem AWS CLI vorinstalliert.	Standardmäßig werden die Amazon Linux 2-basierten Amazon-ECS-optimierten AMIs (Amazon-ECS-optimiertes Amazon Linux 2 AMI, Amazon-ECS-optimiertes Amazon Linux 2 (arm64) AMI und Amazon ECS GPU-optimiertes AMI) mit einem einzigen 30-Gib Root-Volume geliefert. Sie können die Größe des 30-GiB-Root-Volumes zum Startzeitpunkt ändern, um den verfügbaren Speicherplatz auf Ihrer Container-Instance zu erhöhen. Dieser Speicher wird für das Betriebssystem sowie für Docker-Images und Metadaten verwendet. Das Standard-Dateisystem für das Amazon-ECS-optimierte Amazon Linux 2-AMI ist `xfs` und Docker verwendet
	Amazon-ECS-optimiertes Amazon-Linux-2-AMI	Dies ist für Ihre Amazon-ECS-Workloads. Das Amazon-ECS-optimierte Amazon Linux 2-AMI ist nicht mit dem AWS CLI vorinstalliert.	
Amazon Linux 2 (arm64)	Amazon-ECS-optimiertes Amazon Linux 2 (arm64) AMI mit Kernel 5.10	Dieses auf Amazon Linux 2 basierende AMI ist für Ihre Amazon EC2 EC2-Instances vorgesehen, die von ARM-basierten AWS Graviton/Graviton 2-Prozessoren angetrieben	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
		werden und Sie für Ihre Amazon ECS-Workloads den Linux-Kernel 5.10 anstelle von Linux-Kernel 4.14 verwenden möchten. Weitere Informationen finden Sie unter General Purpose Instances im Amazon EC2 EC2-Benutzerhandbuch. Das Amazon ECS-optimierte Amazon Linux 2 (arm64) AMI ist nicht vorinstalliert. AWS CLI	den `overlay2`-Speichertreiber. Weitere Informationen finden Sie unter Verwenden des OverlayFS-Speichertreibers in der Docker-Dokumentation.

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
	Amazon-ECS-optimiertes Amazon Linux 2 (arm64) AMI	Dieses AMI basiert auf Amazon Linux 2 und dient zum Starten Ihrer Amazon EC2 EC2-Instances, die auf ARM-basierten AWS Graviton/Graviton 2-Prozessoren basieren, für Ihre Amazon ECS-Workloads. Das Amazon ECS-optimierte Amazon Linux 2 (arm64) AMI ist nicht vorinstalliert. AWS CLI	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
Amazon Linux 2 (GPU)	GPU-optimierter Kernel 5.10 AMI von Amazon ECS	Dieses AMI basiert auf Amazon Linux 2 und wird für den Start Ihrer Amazon EC2 EC2-GPU-basierten Instances mit Linux-Kernel 5.10 für Ihre Amazon ECS-Workloads empfohlen. Es wird mit NVIDIA-Kernel-Treibern und einer Docker GPU-Laufzeitumgebung geliefert, die die Vorteile von GPUs auf Amazon ECS nutzt. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
	Amazon-ECS-GPU-optimiertes AMI	Dieses AMI basiert auf Amazon Linux 2 und wird für den Start Ihrer Amazon EC2 EC2-GPU-basierten Instances mit Linux-Kernel 4.14 für Ihre Amazon ECS-Workloads empfohlen. Es wird mit NVIDIA-Kernel-Treibern und einer Docker GPU-Laufzeitumgebung geliefert, die die Vorteile von GPUs auf Amazon ECS nutzt. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
Amazon Linux 2 (Neuron)	Amazon ECS hat Amazon Linux 2 (Neuron) Kernel 5.10 AMI optimiert	Basierend auf Amazon Linux 2 ist dieses AMI für Amazon-EC2-Inf1-, Trn1- oder Inf2-Instances bestimmt. Es ist mit AWS Inferentia mit Linux-Kernel 5.10- und AWS Trainium-Treibern sowie der AWS Neuron-Runtime für Docker vorkonfiguriert, wodurch die Ausführung von Inferenz-Workloads für maschinelles Lernen auf Amazon ECS vereinfacht wird. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für AWS Neuron-Workloads für maschinelles Lernen. Das für Amazon ECS optimierte Amazon Linux 2 (Neuron) AMI ist nicht AWS CLI vorinstalliert.	

Betriebssystem	AMI	Beschreibung	Speicherkonfiguration
	Amazon-ECS-optimiertes AMI für Amazon Linux 2 (Neuron)	Basierend auf Amazon Linux 2 ist dieses AMI für Amazon-EC2-Inf1-, Trn1- oder Inf2-Instances bestimmt. Es ist mit den Treibern AWS Inferentia und AWS Trainium sowie der AWS Neuron-Runtime für Docker vorkonfiguriert, wodurch die Ausführung von Inferenz-Workloads für maschinelles Lernen auf Amazon ECS vereinfacht wird. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für AWS Neuron-Workloads für maschinelles Lernen. Das für Amazon ECS optimierte Amazon Linux 2 (Neuron) AMI ist nicht AWS CLI vorinstalliert.	

Amazon ECS stellt ein Changelog für die Linux-Variante des Amazon ECS-optimierten AMI on bereit. GitHub Weitere Informationen finden Sie unter Änderungsprotokoll.

Die Linux-Varianten des für Amazon ECS optimierten AMI verwenden das Amazon Linux 2 AMI oder Amazon Linux 2023 AMI als Basis. Sie können den Quellnamen des AMI für Amazon Linux 2 oder den Namen des AMI für Amazon Linux 2023 für jede Variante abrufen, indem Sie die API von Systems Manager Parameter Store abfragen. Weitere Informationen finden Sie unter Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten. Die AMI-Versionshinweise von Amazon Linux 2 sind ebenfalls verfügbar. Weitere Informationen finden Sie in den Versionshinweisen zu Amazon Linux 2. Die Versionshinweise von Amazon Linux 2023 sind ebenfalls verfügbar. Weitere Informationen finden Sie in den Versionshinweisen zu Amazon Linux 2023.

Auf den folgenden Seiten finden Sie weitere Informationen zu den Änderungen:

- Quelle: AMI-Versionshinweise zu GitHub
- Versionshinweise für die Docker-Engine in der Docker-Dokumentation
- NVIDIA-Treiberdokumentation in der NVIDIA-Dokumentation
- Amazon ECS-Agent-Änderungsprotokoll aktiviert GitHub

  Der Quellcode für die `ecs-init`-Anwendung sowie die Skripts und die Konfiguration für das Paketieren des Agenten sind jetzt Teil des Agent-Repositorys. Ältere Versionen von `ecs-init` und deren Verpackung finden Sie im Amazon ecs-init Changelog unter GitHub

Anwenden von Sicherheitsupdates auf das Amazon ECS-optimierte AMI

Die Amazon ECS-optimierten AMIs, die auf Amazon Linux basieren, enthalten eine angepasste Version von. cloud-init Cloud-initist ein Paket, das zum Bootstrapping von Linux-Images in einer Cloud-Computing-Umgebung und zum Ausführen der gewünschten Aktionen beim Starten einer Instance verwendet wird. Standardmäßig werden bei allen Amazon ECS-optimierten AMIs, die auf Amazon Linux basieren und vor dem 12. Juni 2024 veröffentlicht wurden, alle Sicherheitsupdates der Kategorien „Kritisch“ und „Wichtig“ beim Start der Instance angewendet.

Ab den Versionen der Amazon ECS-optimierten AMIs, die auf Amazon Linux 2 basieren, vom 12. Juni 2024 umfasst das Standardverhalten nicht mehr das Aktualisieren von Paketen beim Start. Stattdessen empfehlen wir Ihnen, auf ein neues Amazon ECS-optimiertes AMI zu aktualisieren, sobald Versionen verfügbar sind. Die Amazon ECS-optimierten AMIs werden veröffentlicht, wenn Sicherheitsupdates oder grundlegende AMI-Änderungen verfügbar sind. Dadurch wird sichergestellt, dass Sie die neuesten Paketversionen und Sicherheitsupdates erhalten und dass die Paketversionen bei Instance-Starts unveränderlich sind. Weitere Informationen zum Abrufen des neuesten Amazon ECS-optimierten AMI finden Sie unter. Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten

Wir empfehlen, Ihre Umgebung so zu automatisieren, dass sie auf ein neues AMI aktualisiert wird, sobald diese verfügbar sind. Informationen zu den verfügbaren Optionen finden Sie unter Amazon ECS ermöglicht eine einfachere EC2-Kapazitätsverwaltung mit Managed Instance Draining.

Um weiterhin „Kritische“ und „Wichtige“ Sicherheitsupdates manuell auf eine AMI-Version anzuwenden, können Sie den folgenden Befehl auf Ihrer Amazon EC2 EC2-Instance ausführen.

```
yum update --security
```

Wenn Sie Sicherheitsupdates beim Start wieder aktivieren möchten, können Sie beim Starten Ihrer Amazon EC2 EC2-Instance die folgende Zeile zum `#cloud-config` Abschnitt der Cloud-Init-Benutzerdaten hinzufügen. Weitere Informationen finden Sie unter Using cloud-init on Amazon Linux 2 im Amazon Linux-Benutzerhandbuch.

```
#cloud-config
repo_upgrade: security
```

Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten

Sie können die Amazon ECS-optimierten AMI-Metadaten programmgesteuert abrufen. Die Metadaten umfassen den AMI-Namen, die Amazon ECS-Container-Agent-Version und die Amazon ECS-Laufzeitversion, die die Docker-Version enthält.

Wenn Sie mit der Konsole einen Cluster erstellen, erstellt Amazon ECS eine Startvorlage für Ihre Instances mit dem neuesten AMI, das dem ausgewählten Betriebssystem zugeordnet ist.

Wenn Sie AWS CloudFormation einen Cluster erstellen, ist der SSM-Parameter Teil der Amazon EC2 EC2-Startvorlage für die Auto Scaling Scaling-Gruppeninstanzen. Sie können die Vorlage so konfigurieren, dass ein dynamischer Systems Manager Manager-Parameter verwendet wird, um zu bestimmen, welches Amazon ECS-optimierte AMI bereitgestellt werden soll. Dieser Parameter stellt sicher, dass der Stack bei jeder Bereitstellung überprüft, ob ein Update verfügbar ist, das auf die EC2-Instances angewendet werden muss. Ein Beispiel für die Verwendung des Systems Manager Manager-Parameters finden Sie unter Erstellen eines Amazon ECS-Clusters mit dem Amazon ECS-optimierten Amazon Linux 2023 AMI im AWS CloudFormation Benutzerhandbuch.

Die AMI-ID, der Image-Name, das Betriebssystem, die Container-Agent-Version, Quell-Image-Name und die Laufzeitversion für jede Variante der Amazon-ECS-optimierten AMIs können programmgesteuert abgerufen werden, indem Sie die Parameterspeicher-API von Systems Manager abfragen. Weitere Informationen zur Systems Manager Parameter Store-API finden Sie unter GetParametersund GetParametersByPath.

> **Note**
>
> Ihr administrativer Benutzer muss über die folgenden IAM-Berechtigungen verfügen, um die Amazon-ECS-optimierten AMI-Metadaten abzurufen. Diese Berechtigungen wurden der `AmazonECS_FullAccess`-IAM-Richtlinie hinzugefügt.
>
> - ssm: GetParameters
> - ssm: GetParameter
> - ssm: GetParameters ByPath

Systems Manager Parameterspeicher-Parameterformat

Im Folgenden ist das Format des Parameternamens für jede Amazon-ECS-optimierte AMI-Variante aufgeführt.

Linux Amazon-ECS-optimierte AMIs

- AMI-Metadaten für Amazon Linux 2023:

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/<version>
```

- AMI-Metadaten für Amazon Linux 2023 (arm64):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/arm64/<version>
```

- AMI-Metadaten für Amazon Linux 2023 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/<version>
```

- Amazon Linux 2-AMI Metadaten:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/<version>
```

- Amazon Linux 2 AMI-Metadaten mit Kernel 5.10:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/<version>
```

- Amazon Linux 2 (arm64) AMI-Metadaten:

```
/aws/service/ecs/optimized-ami/amazal2023neuronamion-linux-2/arm64/<version>
```

- Amazon Linux 2 (arm64) AMI-Metadaten mit Kernel 5.10:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/arm64/<version>
```

- GPU-optimierte Amazon ECS Kernel-5.10-AMI-Metadaten:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/gpu/<version>
```

- Amazon Linux 2 (GPU) AMI-Metadaten:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/<version>
```

- Amazon ECS-optimierte Amazon Linux 2 (Neuron) -Kernel 5.10 AMI-Metadaten:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/inf/<version>
```

- AMI-Metadaten für Amazon Linux 2 (Neuron):

```
/aws/service/ecs/optimized-ami/amazon-linux-2/inf/<version>
```

Das folgende Parameternamensformat ruft die Image-ID des neuesten stabilen Amazon-ECS-optimierten AMI für Linux 2 mit dem Sub-Parameter `image_id` ab.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id
```

Das folgende Parameternamen-Format ruft die Metadaten einer bestimmten Amazon-ECS-optimierten AMI-Version ab, indem der AMI-Name angegeben wird.

- Amazon-ECS-optimierte Amazon Linux 2-AMI Metadaten:

```
/aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20181112-x86_64-ebs
```

> **Note**
>
> Alle Versionen des Amazon-ECS-optimierten Amazon Linux 2-AMI stehen zum Abrufen bereit. Nur Amazon-ECS-optimierte AMI-Versionen `amzn-ami-2017.09.l-amazon-ecs-optimized` (Linux) und höher können abgerufen werden.

Beispiele

Die folgenden Beispiele zeigen, wie Sie die Metadaten für jede Amazon-ECS-optimierte AMI-Variante abrufen können.

Abrufen der Metadaten des neuesten stabilen Amazon-ECS-optimierten AMI

Sie können das neueste stabile Amazon ECS-optimierte AMI AWS CLI mit den folgenden AWS CLI Befehlen abrufen.

Linux Amazon-ECS-optimierte AMIs

- Für die Amazon-ECS-optimierten AMIs von Amazon Linux 2023:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Für die Amazon-ECS-optimierten AMIs für Amazon Linux 2023 (arm64):

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Für die Amazon ECS-optimierten Amazon Linux 2023 (Neuron) AMIs:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
neuron/recommended --region us-east-1
```

- Für die Amazon-ECS-optimierten Amazon-Linux-2-AMIs mit Kernel 5.10:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Für die Amazon-ECS-optimierten Amazon Linux 2-AMIs:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Für die Amazon-ECS-optimierten Amazon-Linux-2-AMIs mit Kernel 5.10 (arm64):

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Für die Amazon-ECS-optimierten Amazon Linux 2 (arm64) AMIs:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazal2023neuronamion-
linux-2/arm64/recommended --region us-east-1
```

- Für die GPU-optimierten Kernel-5.10-AMIs von Amazon ECS:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Für die für Amazon ECS GPU-optimierten AMIs:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Für die Amazon ECS-optimierten Amazon Linux 2 (Neuron) -Kernel 5.10-AMIs:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/inf/recommended --region us-east-1
```

- Amazon-ECS-optimiertes AMI für Amazon Linux 2 (Neuron):

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended --region us-east-1
```

Abrufen der Image-ID des neuesten empfohlenen Amazon-ECS-optimierten AMIs für Amazon Linux 2023

Sie können die Image-ID der aktuellen empfohlenen Amazon-ECS-optimierten AMI-ID für Amazon Linux 2023 mit dem Sub-Parameter `image_id` abrufen.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-
linux-2023/recommended/image_id --region us-east-1
```

Um nur den `image_id`-Wert abzurufen, können Sie den spezifischen Parameterwert abzufragen, z. B.:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Abrufen der Metadaten einer bestimmten Amazon-ECS-optimierten Amazon Linux 2-AMI-Version

Rufen Sie mit dem folgenden AWS CLI Befehl die Metadaten einer bestimmten Amazon ECS-optimierten Amazon Linux AWS CLI AMI-Version ab. Ersetzen Sie den AMI-Namen durch den Namen des abzurufenden Amazon-ECS-optimierten Amazon Linux AMI.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-
ecs-hvm-2.0.20200928-x86_64-ebs --region us-east-1
```

Abrufen der Amazon ECS-optimierten Amazon Linux 2-Kernel-5.10-AMI-Metadaten mithilfe der Systems Manager Manager-API GetParametersByPath

Rufen Sie die Amazon ECS-optimierten Amazon Linux 2-AMI-Metadaten mit der Systems Manager GetParametersByPath Manager-API ab, indem Sie den AWS CLI folgenden Befehl verwenden.

```
aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/ --region us-east-1
```

Abrufen der Image-ID des neuesten empfohlenen Amazon ECS-optimierten Amazon Linux 2-Kernel-5.10-AMI

Sie können die Image-ID der neuesten empfohlenen Amazon ECS-optimierten Amazon Linux 2 Kernel 5.10 AMI-ID mithilfe des Unterparameters abrufen. `image_id`

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended/image_id --region us-east-1
```

Um nur den `image_id`-Wert abzurufen, können Sie den spezifischen Parameterwert abzufragen, z. B.:

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Verwenden des neuesten empfohlenen Amazon ECS-optimierten AMI in einer Vorlage AWS CloudFormation

Sie können auf das neueste empfohlene Amazon-ECS-optimierte AMI in einer AWS CloudFormation -Vorlage verweisen, indem Sie auf den Systems Manager Parameterspeichernamen verweisen.

Linux-Beispiel

```
Parameters:kernel-5.10
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/
image_id
```

Amazon-ECS-optimiertes Linux-AMI-Entwicklungsskript

Amazon ECS hat die Entwicklungsskripts, die zum Erstellen der Linux-Varianten des Amazon-ECS-optimierten AMI verwendet werden, als Open Source bereitgestellt. Diese Build-Skripte sind jetzt auf verfügbar. GitHub Weitere Informationen finden Sie unter amazon-ecs-ami unter. GitHub

Wenn Sie das Amazon ECS-optimierte AMI anpassen müssen, finden Sie weitere Informationen unter Amazon ECS Optimized AMI Build Recipies auf. GitHub

Das Build-Skripts-Repository enthält eine HashiCorpPacker-Vorlage und Build-Skripte zur Generierung der einzelnen Linux-Varianten des Amazon ECS-optimierten AMI. Diese Skripts sind die Informationsquelle für Amazon ECS-optimierte AMI-Builds, sodass Sie dem GitHub Repository folgen können, um Änderungen an unseren AMIs zu überwachen. Beispielsweise möchten Sie vielleicht, dass Ihr eigenes AMI dieselbe Version von Docker nutzt, die das Amazon-ECS-Team für das offizielle AMI verwendet.

Weitere Informationen finden Sie im Amazon ECS AMI-Repository unter aws/amazon-ecs-ami am. GitHub

So erstellen Sie ein Amazon-ECS-optimiertes Linux-AMI

1. `aws/amazon-ecs-ami` GitHub Klonen Sie das Repo.

```
git clone https://github.com/aws/amazon-ecs-ami.git
```

2. Fügen Sie eine Umgebungsvariable für die AWS Region hinzu, die bei der Erstellung des AMI verwendet werden soll. Ersetzen Sie den `us-west-2`-Wert durch die zu verwendende Region.

```
export REGION=us-west-2
```

3. Zur Entwicklung des AMIs wird ein Makefile bereitgestellt. Verwenden Sie aus dem Stammverzeichnis des geklonten Repositorys einen der folgenden Befehle, der der Linux-Variante des Amazon-ECS-optimierten AMI entspricht, das Sie erstellen möchten.

   - Amazon-ECS-optimiertes Amazon Linux 2-AMI

     ```
     make al2
     ```

   - Amazon-ECS-optimiertes Amazon Linux 2 (arm64) AMI

     ```
     make al2arm
     ```

   - Amazon-ECS-GPU-optimiertes AMI

     ```
     make al2gpu
     ```

   - Amazon-ECS-optimiertes AMI für Amazon Linux 2 (Neuron)

     ```
     make al2inf
     ```

   - Amazon-ECS-optimiertes AMI für Amazon Linux 2023

     ```
     make al2023
     ```

   - Amazon-ECS-optimiertes AMI für Amazon Linux 2023 (arm64)

     ```
     make al2023arm
     ```

   - Amazon-ECS-optimiertes AMI für Amazon Linux 2023 (Neuron)

     ```
     make al2023neu
     ```

Amazon-ECS-optimierte Bottlerocket-AMIs

Bottlerocketist ein Linux Open-Source-Betriebssystem, das speziell AWS für den Betrieb von Containern auf virtuellen Maschinen oder Bare-Metal-Hosts entwickelt wurde. Das Amazon-ECS-optimierte Bottlerocket-AMI ist sicher und enthält nur die Mindestanzahl an Paketen, die zum Ausführen von Containern erforderlich ist. Dies verbessert die Ressourcennutzung, reduziert die Angriffsfläche für Sicherheitsangriffe und trägt zur Senkung des Verwaltungsaufwands bei. Das Bottlerocket-AMI ist auch in Amazon ECS integriert, um den betrieblichen Aufwand zu reduzieren, der mit der Aktualisierung von Container-Instances in einem Cluster verbunden ist.

Bottlerocket unterscheidet sich auf folgende Weise von Amazon Linux:

- Bottlerocket enthält keinen Paketmanager und die zugehörige Software kann nur als Container ausgeführt werden. Aktualisierungen von Bottlerocket werden in einem einzigen Schritt durchgeführt und können wieder rückgängig gemacht werden, wodurch die Wahrscheinlichkeit von Aktualisierungsfehlern verringert wird.
- Der primäre Mechanismus zur Verwaltung von Bottlerocket-Hosts ist ein Container-Scheduler. Im Gegensatz zu Amazon Linux ist die Anmeldung bei einzelnen Bottlerocket-Instances nur selten und nur für erweiterte Debugging- und Fehlerbehebungszwecke vorgesehen.

Weitere Informationen zu Bottlerocket finden Sie in der Dokumentation und den Releases auf GitHub.

Es gibt Varianten des Amazon ECS-optimierten Bottlerocket AMI für Kernel 6.1 und Kernel 5.10.

Die folgenden Varianten verwenden Kernel 6.1:

- `aws-ecs-2`
- `aws-ecs-2-nvidia`

Die folgenden Varianten verwenden Kernel 5.1.10:

- `aws-ecs-1`
- `aws-ecs-1-nvidia`

  Weitere Informationen über die `aws-ecs-1-nvidia`-Variante finden Sie unter Ankündigung der NVIDIA-GPU-Unterstützung für Bottlerocket auf Amazon ECS.

Überlegungen

Beachten Sie Folgendes, wenn Sie ein Bottlerocket-AMI mit Amazon ECS verwenden.

- Bottlerocket unterstützt Amazon-EC2-Instances mit `x86_64`- und `arm64`-Prozessoren. Die Bottlerocket-AMI wird nicht für die Verwendung mit Amazon-EC2-Instances mit einem Inferentia-Chip empfohlen.
- Bottlerocket-Images enthalten keine SSH-Server oder Shell. Sie können jedoch out-of-band Verwaltungstools verwenden, um SSH-Administratorzugriff zu erhalten und Bootstrapping durchzuführen. Weitere Informationen finden Sie in diesen Abschnitten im bottlerocket README.md auf GitHub:
  - Exploration (Erkundung)
  - Administrator-Container
- Standardmäßig hat Bottlerocket einen Steuerungs-Container, der aktiviert ist. Dieser Container führt den AWS Systems Manager -Agenten aus, den Sie verwenden können, um Befehle auszuführen oder Shell-Sitzungen auf Amazon-EC2-Bottlerocket-Instances zu starten. Weitere Informationen finden Sie unter Session Manager einrichten im AWS Systems Manager - Benutzerhandbuch.

- Bottlerocket ist für Container-Workloads optimiert und konzentriert sich auf Sicherheit. Bottlerocket enthält keinen Paketmanager und ist unveränderlich. Informationen zu den Sicherheitsfunktionen und Anleitungen finden Sie unter Sicherheitsfunktionen und Sicherheitsleitlinien am. GitHub
- Der `awsvpc`-Netzwerkmodus wird für Bottlerocket AMI Version `1.1.0` oder höher unterstützt.
- App Mesh in einer Aufgabendefinition wird für die Bottlerocket-AMI-Version `1.15.0` oder höher unterstützt.
- Der `initProcessEnabled` Aufgabendefinitionsparameter wird für die Bottlerocket AMI-Version `1.19.0` oder höher unterstützt.
- Die Bottlerocket-AMIs unterstützen auch die folgenden Services und Features nicht:
  - ECS Anywhere
  - Service Connect
  - Amazon EFS im verschlüsselten Modus und `awsvpc`-Netzwerkmodus
  - Elastic-Inference-Beschleuniger

Abrufen von Amazon Bottlerocket ECS-optimierten AMI-Metadaten

Sie können die Amazon Machine Image (AMI) -ID für Amazon ECS-optimierte AMIs abrufen, indem Sie die AWS Systems Manager Parameter Store-API abfragen. Mit diesem Parameter müssen Sie Amazon-ECS-optimierte AMI-IDs nicht manuell nachschlagen. Weitere Informationen zur Systems Manager Parameter Store-API finden Sie unter GetParameter. Der von Ihnen verwendete Benutzer muss über die `ssm:GetParameter`-IAM-Berechtigung zum Abrufen der Amazon-EKS-optimierten AMI-Metadaten verfügen.

`aws-ecs-2`Bottlerocket AMI-Variante

Sie können die neueste stabile `aws-ecs-2` Bottlerocket-AMI-Variante nach AWS-Region und Architektur mit dem oder dem AWS CLI abrufen. AWS Management Console

- AWS CLI— Sie können die Image-ID des neuesten empfohlenen Amazon ECS-optimierten Bottlerocket AMIs mit dem folgenden AWS CLI Befehl abrufen, indem Sie den Unterparameter verwenden. `image_id` Ersetzen Sie den *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub. Um eine andere Version als die neueste abzurufen, ersetzen Sie`latest` mit der Versionsnummer.
  - Für die 64-Bit-(`x86_64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Für die 64 Bit Arm (`arm64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console – Sie können die empfohlene Amazon-ECS-optimierte AMI-ID mithilfe einer URL in der AWS Management Console abfragen. Die URL öffnet die Amazon-EC2-Systems-Manager-Konsole mit dem Wert der ID für den Parameter. Ersetzen Sie in der folgenden URL *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub.
  - Für die 64-Bit-(`x86_64`)-Architektur:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id/description?region=region#
    ```

- Für die 64 Bit Arm (arm64)-Architektur:

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
aws-ecs-2/arm64/latest/image_id/description?region=region#
```

aws-ecs-2-nvidia Bottlerocket AMI-Variante

Sie können die neueste stabile aws-ecs-2-nvdia Bottlerocket-AMI-Variante nach Region und Architektur mit dem oder dem AWS CLI abrufen. AWS Management Console

- AWS CLI— Sie können die Image-ID des neuesten empfohlenen Amazon ECS-optimierten Bottlerocket AMIs mit dem folgenden AWS CLI Befehl abrufen, indem Sie den Unterparameter verwenden. image_id Ersetzen Sie den *region* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub. Um eine andere Version als die neueste abzurufen, ersetzen Sielatest mit der Versionsnummer.
  - Für die 64-Bit-(x86_64)-Architektur:

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
ecs-2-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
```

  - Für die 64 Bit Arm (arm64)-Architektur:

```
aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
ecs-2-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
```

- AWS Management Console – Sie können die empfohlene Amazon-ECS-optimierte AMI-ID mithilfe einer URL in der AWS Management Console abfragen. Die URL öffnet die Amazon-EC2-Systems-Manager-Konsole mit dem Wert der ID für den Parameter. Ersetzen Sie in der folgenden URL *region* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub.
  - Für die 64 Bit (x86_64)-Architektur:

```
https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id/description?region=region#
```

  - Für die 64 Bit Arm (arm64)-Architektur:

```
https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id/description?region=region#
```

`aws-ecs-1`Bottlerocket AMI-Variante

Sie können die neueste stabile `aws-ecs-1` Bottlerocket-AMI-Variante nach AWS-Region und Architektur mit dem oder dem AWS CLI abrufen. AWS Management Console

- AWS CLI— Sie können die Image-ID des neuesten empfohlenen Amazon ECS-optimierten Bottlerocket AMIs mit dem folgenden AWS CLI Befehl abrufen, indem Sie den Unterparameter verwenden. `image_id` Ersetzen Sie den *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub. Um eine andere Version als die neueste abzurufen, ersetzen Sie`latest` mit der Versionsnummer.
  - Für die 64-Bit-(`x86_64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Für die 64 Bit Arm (`arm64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console – Sie können die empfohlene Amazon-ECS-optimierte AMI-ID mithilfe einer URL in der AWS Management Console abfragen. Die URL öffnet die Amazon-EC2-Systems-Manager-Konsole mit dem Wert der ID für den Parameter. Ersetzen Sie in der folgenden URL *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub.
  - Für die 64-Bit-(`x86_64`)-Architektur:

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-1/x86_64/latest/image_id/description
    ```

  - Für die 64 Bit Arm (`arm64`)-Architektur:

```
https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-1/arm64/latest/image_id/description
```

`aws-ecs-1-nvidia`Bottlerocket AMI-Variante

Sie können die neueste stabile `aws-ecs-1-nvdia` Bottlerocket-AMI-Variante nach Region und Architektur mit dem oder dem AWS CLI abrufen. AWS Management Console

- AWS CLI— Sie können die Image-ID des neuesten empfohlenen Amazon ECS-optimierten Bottlerocket AMIs mit dem folgenden AWS CLI Befehl abrufen, indem Sie den Unterparameter verwenden. `image_id` Ersetzen Sie den *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub. Um eine andere Version als die neueste abzurufen, ersetzen Sie`latest` mit der Versionsnummer.
  - Für die 64-Bit-(`x86_64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Für die 64 Bit Arm (`arm64`)-Architektur:

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console – Sie können die empfohlene Amazon-ECS-optimierte AMI-ID mithilfe einer URL in der AWS Management Console abfragen. Die URL öffnet die Amazon-EC2-Systems-Manager-Konsole mit dem Wert der ID für den Parameter. Ersetzen Sie in der folgenden URL *`region`* durch den Regionalcode, für den Sie die AMI-ID benötigen. Informationen zu den unterstützten finden Sie AWS-Regionen unter Finding an AMI on GitHub.
  - Für die 64 Bit (`x86_64`)-Architektur:

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Für die 64 Bit Arm (`arm64`)-Architektur:

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
aws-ecs-1-nvidia/arm64/latest/image_id/description?region=region#
```

Nächste Schritte

Ein ausführliches Tutorial zu den ersten Schritten mit dem Bottlerocket Betriebssystem auf Amazon ECS finden Sie unter Using a Bottlerocket AMI with Amazon ECS on GitHub und Getting started with Bottlerocket and Amazon ECS auf der AWS Blogseite.

Informationen zum Starten einer Bottlerocket-Instance finden Sie unter Starten einer Bottlerocket Instance für Amazon ECS

Starten einer Bottlerocket Instance für Amazon ECS

Sie können eine Bottlerocket Instance starten, damit Sie Ihre Container-Workloads ausführen können.

Sie können die verwenden AWS CLI , um die Bottlerocket Instance zu starten.

1. Erstellen Sie eine Datei mit dem Namen `userdata.toml`. Diese Datei wird für Instance-Benutzerdaten verwendet. Ersetzen Sie *`cluster-name`* durch den Namen Ihres Clusters.

   ```
   [settings.ecs]
   cluster = "cluster-name"
   ```

2. Verwenden Sie einen der Befehle, die in the section called “Abrufen von Amazon Bottlerocket ECS-optimierten AMI-Metadaten” enthalten sind, um die Bottlerocket-AMI-ID abzurufen. Sie verwenden dies im folgenden Schritt.
3. Führen Sie den folgenden Befehl aus, um die Bottlerocket-Instance zu starten. Denken Sie daran, die folgenden Parameter zu ersetzen:

   - Ersetzen Sie das *`Subnetz`* durch die ID des privaten oder öffentlichen Subnetzes, in dem Ihre Instance gestartet wird.
   - Ersetzen Sie *`bottlerocket_ami`* durch die AMI-ID aus dem vorherigen Schritt.
   - Ersetzen Sie *`t3.large`* durch den Instance-Typ, den Sie verwenden möchten.
   - Ersetzen Sie *`Region`* durch den Regionscode.

```
aws ec2 run-instances --key-name ecs-bottlerocket-example \
   --subnet-id subnet \
   --image-id bottlerocket_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications
 'ResourceType=instance,Tags=[{Key=bottlerocket,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob die Container-Instance im Cluster registriert ist. Denken Sie beim Ausführen dieses Befehls daran, die folgenden Parameter zu ersetzen:

   - Ersetzen Sie *cluster* durch Ihren Cluster-Namen.
   - Ersetzen Sie *region* durch Ihren Regionalcode.

   ```
   aws ecs list-container-instances --cluster cluster-name --region region
   ```

Eine ausführliche Anleitung zu den ersten Schritten mit dem Bottlerocket Betriebssystem auf Amazon ECS finden Sie unter Using a Bottlerocket AMI with Amazon ECS on and Getting started with Amazon ECS on GitHub und Getting started with Bottlerocketand Amazon ECS auf der Blogseite. AWS

Verwaltung von Amazon ECS Linux-Container-Instances

Wenn Sie EC2-Instances für Ihre Amazon ECS-Workloads verwenden, sind Sie für die Wartung der Instances verantwortlich

Verwaltungsverfahren

- Starten einer Amazon ECS Linux-Container-Instance
- Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe
- Konfiguration von Amazon ECS-Linux-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen
- Ausführen eines Skripts beim Starten einer Amazon ECS Linux-Container-Instance

- Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances
- Reservieren von Amazon ECS Linux-Container-Instance-Speicher
- Fernverwaltung von Amazon ECS-Container-Instances mithilfe von AWS Systems Manager
- Verwenden eines HTTP-Proxys für Amazon ECS Linux-Container-Instances
- Konfiguration vorinitialisierter Instances für Ihre Amazon ECS Auto Scaling Scaling-Gruppe
- Überprüfen des Amazon-ECS-Container-Agenten

Jede Amazon-ECS-Container-Agenten-Version unterstützt unterschiedliche Features und bietet Fehlerbehebungen aus früheren Versionen. Wir empfehlen grundsätzlich, die neueste Version des Amazon-ECS-Container-Agenten zu verwenden, wenn dies möglich ist. Die neueste Version für Ihren Container-Agenten finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Unter https://github.com/aws/amazon-ecs-agent/releases finden Sie alle Features und Verbesserungen jeder Agenten-Version.

> **Important**
>
> Die Mindestversion von Docker für zuverlässige Metriken ist die Docker-Version `v20.10.13` und neuer, die in Amazon-ECS-optimiertem AMI `20220607` und neuer enthalten ist.
> Die Amazon-ECS-Agent-Versionen `1.20.0` und neuer haben die Unterstützung für Docker-Versionen älter als `1.9.0` eingestellt.

Starten einer Amazon ECS Linux-Container-Instance

Sie können Amazon ECS-Container-Instances mithilfe der Amazon EC2 EC2-Konsole erstellen.

Sie können eine Instance mit verschiedenen Methoden starten, einschließlich der Amazon EC2 EC2-Konsole und dem SDK. AWS CLI Informationen zu den anderen Methoden zum Starten einer Instance finden Sie unter Starten Ihrer Instance im Amazon EC2 EC2-Benutzerhandbuch.

Weitere Informationen zum Startassistenten finden Sie unter Starten einer Instance mithilfe des Assistenten zum Starten einer Instance im Amazon EC2 EC2-Benutzerhandbuch.

Bevor Sie beginnen, führen Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS aus.

Sie können den neuen Amazon-EC2-Assistenten verwenden, um eine Instance zu starten. Der Launch Instance Wizard gibt alle Startparameter an, die zum Starten einer Instance erforderlich sind.

Parameter für die Instance-Konfiguration

- Verfahren
- Name und Tags
- Anwendungs- und Betriebssystem-Images (Amazon Machine Image)
- Instance-Typ
- Schlüsselpaar (Anmeldung)
- Network settings (Netzwerkeinstellungen)
- Speicher konfigurieren
- Erweiterte Details

Verfahren

Bevor Sie beginnen, führen Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS aus.

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. In der Navigationsleiste oben auf dem Bildschirm wird die aktuelle AWS Region angezeigt (z. B. USA Ost (Ohio)). Wählen Sie eine Region aus, in der die Instance gestartet werden soll.
3. Wählen Sie im Dashboard der Amazon EC2-Konsole die Option Instance starten aus.

Name und Tags

Der Instance-Name ist ein Tag, wobei der Schlüssel Name ist und es sich bei dem Wert um den von Ihnen angegebenen Namen handelt. Sie können Instance, Volumes und elastische Grafiken markieren. Bei Spot-Instances können Sie nur die Spot-Instance-Anforderung mit Tags (Markierungen) versehen.

Die Angabe eines Instance-Namens und zusätzlicher Tags ist optional.

- Geben Sie unter Name einen beschreibenden Namen für die Instance ein. Wenn Sie keinen Namen angeben, kann die Instance anhand der ID identifiziert werden, die beim Starten der Instance automatisch generiert wird.
- Wenn Sie zusätzliche Tags hinzufügen möchten, wählen Sie Add additional tags (Zusätzliche Tags hinzufügen) aus. Klicken Sie auf Tag hinzufügen, geben Sie dann einen Schlüssel und einen Wert ein und wählen Sie den Ressourcentyp aus, den Sie markieren möchten. Wählen Sie für jedes weitere Tag Add another Tag (Weiteres Tag hinzufügen) aus.

Anwendungs- und Betriebssystem-Images (Amazon Machine Image)

Ein Amazon Machine Image (AMI) enthält die Informationen, die zum Starten einer Instance erforderlich sind. Ein AMI könnte beispielsweise die für die Funktion als Webserver erforderliche Software (z. B. Apache) und Ihre Website enthalten.

Verwenden Sie die Suchleiste, um ein geeignetes Amazon ECS-optimiertes AMI zu finden, das von veröffentlicht wurde. AWS

1. Geben Sie einen der folgenden Begriffe in die Suchleiste ein.
   - **`ami-ecs`**
   - Der Wert eines Amazon-ECS-optimierten AMI.

     Informationen zu den neuesten Amazon-ECS-optimierten AMIs und deren Werten finden Sie unter Linux-Amazon-ECS-optimiertes AMI.
2. Drücken Sie die Eingabetaste.
3. Wählen Sie auf der Seite Amazon Machine Image (AMI) auswählen die Kategorie AWS - Marketplace-AMIs aus.
4. Wählen Sie im Bereich Ergebnisse verfeinern auf der linken Seite Amazon Web Services als Publisher aus.
5. Wählen in der Zeile des AMI, das Sie verwenden wollen, Auswählen aus.

   Wählen Sie andernfalls Abbrechen (oben rechts), um zum Start-Instance-Assistenten zurückzukehren, ohne ein AMI zu wählen. Ein Standard-AMI ist ausgewählt. Stellen Sie sicher, dass das AMI die in Linux-Instances aufgeführten Anforderungen erfüllt.

Instance-Typ

Der Instance-Typ definiert die Hardware-Konfiguration und Größe der Instance. Größere Instance-Typen haben mehr CPU und Arbeitsspeicher. Weitere Informationen finden Sie unter Instance-Typen.

- Wählen Sie unter Instance-Typ den Instance-Typ für die Instance aus.

  Von dem von Ihnen ausgewählten Instance-Typ ist abhängig, welche Ressourcen zur Ausführung Ihrer Aufgaben verfügbar sind.

Schlüsselpaar (Anmeldung)

Wählen Sie für Schlüsselpaarname ein vorhandenes Schlüsselpaar aus oder wählen Sie Neues Schlüsselpaar erstellen, um ein neues zu erstellen.

> **Important**
>
> Wenn Sie die Option Proceed without key pair (Not recommended) (Ohne Schlüsselpaar fortfahren (Nicht empfohlen)) auswählen, können Sie keine Verbindung zur Instance herstellen, es sei denn, Sie wählen ein AMI aus, das entsprechend konfiguriert ist, um Benutzern eine andere Anmeldemöglichkeit zu erlauben.

Network settings (Netzwerkeinstellungen)

Konfigurieren Sie die Netzwerkeinstellungen nach Bedarf.

- Networking platform (Netzwerkplattform): Wählen Sie Virtual Private Cloud (VPC) aus und geben Sie dann im Abschnitt Network interfaces (Netzwerkschnittstellen) das Subnetz an.
- VPC: Wählen Sie eine vorhandene VPC aus, in der die Sicherheitsgruppe erstellt werden soll.
- Subnetz: Sie können eine Instance in einem Subnetz starten, das einer Availability Zone, einer Local Zone, Wavelength-Zone oder einem Outpost zugeordnet ist.

  Um die Instance in einer Availability Zone zu starten, wählen Sie das Subnetz aus, in dem die Instance gestartet werden soll. Um ein neues Subnetz zu erstellen, wählen Sie Create new subnet aus, um die Amazon VPC-Konsole aufzurufen. Wenn Sie fertig sind, kehren Sie zum Launch Instance Wizard zurück und wählen Sie das Symbol zum Aktualisieren, um Ihr Subnetz in die Liste zu laden.

  Um die Instance in einer Local Zone zu starten, wählen Sie ein Subnetz aus, das Sie in der Local Zone erstellt haben.

  Um eine Instance in einem Outpost zu starten, wählen Sie ein Subnetz in einer VPC aus, die Sie dem Outpost zugeordnet haben.
- Auto-assign Public IP (Öffentliche IP automatisch zuweisen): Wenn Ihre Instance über das öffentliche Internet zugänglich sein soll, muss das Feld Auto-assign Public IP (Öffentliche IP automatisch zuweisen) auf Enable (Aktivieren) festgelegt sein. Falls nicht, setzen Sie dieses Feld auf Disable (Deaktivieren).

> **Note**
>
> Container-Instances benötigen Zugriff, um mit dem Amazon-ECS-Service-Endpunkt zu kommunizieren. Dies kann über einen Schnittstellen-VPC-Endpunkt oder über Ihre Container-Instances mit öffentlichen IP-Adressen erfolgen.
> Weitere Informationen zu Schnittstellen-VPC-Endpunkten finden Sie unter Amazon ECS und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).
> Wenn Sie keinen Schnittstellen-VPC-Endpunkt konfiguriert haben und Ihre Container-Instances keine öffentlichen IP-Adressen haben, müssen Sie Netzwerkadressübersetzung (Network Address Translation, NAT) verwenden, um diesen Zugriff zur Verfügung zu stellen. Weitere Informationen finden Sie unter NAT-Gateways im Amazon-VPC-Benutzerhandbuch und unter Verwenden eines HTTP-Proxys für Amazon ECS Linux-Container-Instances in dieser Anleitung.

- Firewall (security groups) (Firewall (Sicherheitsgruppen)): Verwenden Sie eine Sicherheitsgruppe, um Firewallregeln für Ihre Container-Instance zu definieren. Diese Regeln legen fest, welcher eingehende Netzwerkverkehr an Ihre Container-Instance weitergeleitet wird. Der gesamte übrige Datenverkehr wird ignoriert.
  - Um eine vorhandene Sicherheitsgruppe auszuwählen, wählen Sie Select existing security group (Vorhandene Sicherheitsgruppe auswählen) und wählen Sie die Sicherheitsgruppe aus, die Sie in Einrichtung für die Verwendung von Amazon ECS erstellt haben.

Speicher konfigurieren

Die von Ihnen ausgewählte AMI beinhaltet ein oder mehrere Speicher-Volumes, einschließlich eines Root-Volumes. Sie können zusätzliche Volumes angeben, die an die Instance angehängt werden sollen.

Sie können die Ansicht Simple (Einfach) verwenden.

- Storage type (Speichertyp): Konfigurieren Sie den Speicher für Ihre Container-Instance.

  Wenn Sie das Amazon-ECS-optimierte Amazon Linux 2-AMI verwenden, ist für Ihre Instance ein einzelnes 30 GiB-Volume konfiguriert, das zwischen dem Betriebssystem und Docker geteilt wird.

Wenn Sie das Amazon-ECS-optimierte AMI verwenden, erhält Ihre Instance zwei Volumes konfiguriert. Das Root-Volume ist für die Nutzung durch das Betriebssystem und das zweite Amazon EBS-Volume (zugeordnet zu `/dev/xvdcz`) ist für die Nutzung durch Docker vorgesehen.

Sie haben auch die Möglichkeit, die Volume-Größen für Ihre Instance entsprechend den Anforderungen Ihrer Anwendung zu erhöhen oder zu verringern.

Erweiterte Details

Erweitern Sie für Advanced details (Erweiterte Details) den Bereich zur Ansicht der Felder und geben Sie zusätzliche Parameter für die Instance an.

- Purchasing option (Kaufoption): Wählen Sie Request Spot instances (Spot-Instances anfordern) aus, um Spot-Instances anzufordern. Außerdem müssen Sie in den anderen Felder im Zusammenhang mit Spot-Instances Werte festlegen. Weitere Informationen finden Sie unter Spot-Instance-Anforderungen.

  > **Note**
  >
  > Wenn Sie Spot-Instances verwenden und die Meldung `Not available` angezeigt wird, muss möglicherweise ein anderer Instance-Typ angegeben werden.

  .

- IAM instance profile (IAM-Instance-Profil): Wählen Sie die IAM-Rolle Ihrer Container-Instance aus. Diese heißt normalerweise `ecsInstanceRole`.

  > **Important**
  >
  > Wenn Sie Ihre Container-Instance nicht mit den richtigen IAM-Berechtigungen starten, kann Ihr Amazon-ECS-Agent keine Verbindung mir Ihrem Cluster herstellen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

- (Optional) User data (Benutzerdaten): Konfigurieren Sie Ihre Amazon-ECS-Container-Instance mit Benutzerdaten, wie z. B. den Agenten-Umgebungsvariablen aus Konfiguration des Amazon-ECS-Container-Agenten. Amazon EC2-Benutzerdaten-Skripts werden nur einmal ausgeführt, wenn die

Instance erstmals gestartet wird. Im Folgenden finden Sie einige gängige Beispiele dafür, wofür Benutzerdaten verwendet werden:

- Standardmäßig startet Ihre Container-Instance in Ihrem Standard-Cluster. Für einen Start in einem nicht standardmäßigen Cluster wählen Sie die Liste Advanced Details. Fügen Sie dann das folgende Skript in das Feld User data ein und ersetzen Sie *your_cluster_name* durch den Namen Ihres Clusters.

```
#!/bin/bash
echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
```

- Wenn sich eine `ecs.config`-Datei in Amazon S3 befindet und für Amazon S3 schreibgeschützter Zugriff auf Ihre Container-Instance-Rolle aktiviert ist, wählen Sie die Liste Advanced Details (Erweiterte Details). Fügen Sie dann das folgende Skript in das Feld Benutzerdaten ein und ersetzen *Sie your_bucket_name durch den Namen* Ihres Buckets, um die Konfigurationsdatei beim Start zu installieren AWS CLI und zu schreiben.

> **Note**
>
> Weitere Informationen zu dieser Konfiguration finden Sie unter Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

- Geben Sie Tags für Ihre Container-Instance mit dem Konfigurationsparameter `ECS_CONTAINER_INSTANCE_TAGS` an. Dadurch werden Tags erstellt, die nur mit Amazon ECS verknüpft sind, sie können nicht über die Amazon EC2-API aufgelistet werden.

> **Important**
>
> Wenn Sie Ihre Container-Instances mit einer Amazon EC2 Auto Scaling-Gruppe starten, sollten Sie den Agent-Konfigurationsparameter ECS_CONTAINER_INSTANCE_TAGS verwenden, um Tags hinzuzufügen. Dies liegt an der Art und Weise, in der Amazon-EC2-Instances Tags hinzugefügt werden, die mit Auto-Scaling-Gruppen gestartet werden.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Geben Sie Tags für Ihre Container-Instance an und verwenden Sie dann den Konfigurationsparameter `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM`, um sie von Amazon EC2 nach Amazon ECS zu übertragen

  Nachfolgend finden Sie ein Beispiel für ein Benutzerdaten-Skript, das die mit einer Container-Instance verknüpften Tags propagiert und die Container-Instance mit einem Cluster namens `your_cluster_name` registriert:

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=your_cluster_name
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

Weitere Informationen finden Sie unter Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe.

Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe

Wenn Sie eine Amazon EC2 EC2-Instance starten, können Sie Benutzerdaten an die EC2-Instance übergeben. Die Daten können für allgemeine automatisierte Konfigurationsaufgaben und sogar zur Ausführung von Skripts beim Start der Instance verwendet werden. In Amazon ECS werden Benutzerdaten hauptsächlich zur Übergabe von Konfigurationsinformationen an den Docker-Daemon und den Amazon-ECS-Container-Agenten verwendet.

Sie können mehrere Arten von Benutzerdaten an Amazon EC2 übergeben, darunter auch Cloud-Boothooks, Shell-Skripts und `cloud-init`-Anweisungen. Weitere Informationen zu diesen und anderen Formattypen finden Sie in der Cloud-Init-Dokumentation.

Informationen zur Weitergabe der Benutzerdaten bei der Verwendung des Amazon EC2 EC2-Startassistenten finden Sie unterStarten einer Amazon ECS Linux-Container-Instance.

Sie können die Container-Instance so konfigurieren, dass sie Daten in der Container-Agent-Konfiguration oder in der Docker-Daemon-Konfiguration weitergibt.

Amazon-ECS-Container-Agent

Die Linux-Varianten des Amazon-ECS-optimierten AMI suchen beim Start des Containeragenten nach Agenten-Konfigurationsdaten in der Datei `/etc/ecs/ecs.config`. Sie können diese Konfigurationsdaten beim Start mit Amazon EC2-Benutzerdaten angeben. Weitere Informationen zu den verfügbaren Konfigurationsvariablen für den Amazon-ECS-Container-Agenten finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Wenn Sie nur eine einzelne Agent-Konfigurationsvariable wie z. B. den Clusternamen festlegen möchten, kopieren Sie die Variable mit dem Befehl echo in die Konfigurationsdatei:

```
#!/bin/bash
echo "ECS_CLUSTER=MyCluster" >> /etc/ecs/ecs.config
```

Wenn Sie mehrere Variablen in `/etc/ecs/ecs.config` schreiben müssen, verwenden Sie hierfür das folgende `heredoc`-Format. Bei Verwendung dieses Formats wird alles zwischen den Zeilen cat und `EOF` in die Konfigurationsdatei geschrieben.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
ECS_LOGLEVEL=debug
ECS_WARM_POOLS_CHECK=true
EOF
```

Um benutzerdefinierte Instance-Attribute festzulegen, legen Sie die `ECS_INSTANCE_ATTRIBUTES`-Umgebungsvariable fest.

```
#!/bin/bash
cat <<'EOF' >> ecs.config
ECS_INSTANCE_ATTRIBUTES={"envtype":"prod"}
EOF
```

Docker-Daemon

Sie können Docker Daemon-Konfigurationsinformationen mit Amazon EC2-Benutzerdaten angeben. Weitere Informationen zu Konfigurationsoptionen finden Sie in der Docker-Daemon-Dokumentation.

Im folgenden Beispiel werden die benutzerdefinierten Optionen zur Konfigurationsdatei des Docker-Daemons, `/etc/docker/daemon.json`, hinzugefügt, die dann in den Benutzerdaten angegeben wird, wenn die Instance gestartet wird.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"debug": true}
EOF
systemctl restart docker --no-block
```

Im folgenden Beispiel werden die benutzerdefinierten Optionen zur Konfigurationsdatei des Docker-Daemons, `/etc/docker/daemon.json`, hinzugefügt, die dann in den Benutzerdaten angegeben wird, wenn die Instance gestartet wird. Dieses Beispiel zeigt, wie der Docker-Proxy in der Docker-Daemon-Konfigurationsdatei deaktiviert wird.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"userland-proxy": false}
EOF
systemctl restart docker --no-block
```

Konfiguration von Amazon ECS-Linux-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen

Amazon EC2 hält Ihre Spot-Instance an, beendet sie oder versetzt sie in den Ruhezustand, wenn der Spot-Preis den Höchstpreis für Ihre Anforderung überschreitet oder keine Kapazität mehr verfügbar ist. Amazon EC2 stellt eine zweiminütige Spot-Instance-Unterbrechungsmeldung für Beenden- und Anhalten-Aktionen bereit. Es wird nicht die zweiminütige Benachrichtigung für die Ruhezustand bereitgestellt. Wenn das Amazon ECS Spot-Instance-Draining auf der Instance aktiviert ist, erhält Amazon ECS die Benachrichtigung über die Unterbrechung der Spot-Instance und versetzt die Instance in `DRAINING` den Status.

> **Important**
>
> Amazon ECS erhält keine Benachrichtigung von Amazon EC2, wenn Instances durch den Kapazitäts-Neuausgleich des Auto Scalings entfernt werden. Weitere Informationen finden Sie unter Kapazitäts-Neuausgleich von Amazon EC2 Auto Scaling.

Wenn eine Container-Instance auf `DRAINING` festgelegt wird, lässt es Amazon ECS nicht zu, dass die Platzierung neuer Aufgaben in der Container-Instance geplant wird. Serviceaufgaben auf der betroffenen Container-Instance mit dem Status `PENDING` werden umgehend gestoppt. Wenn Container-Instances im Cluster verfügbar sind, werden Ersatzserviceaufgaben darauf gestartet.

Das Spot-Instance-Draining ist standardmäßig ausgeschaltet.

Sie können das Spot-Instance-Draining aktivieren, wenn Sie eine Instance starten. Fügen Sie das folgende Skript in das Feld Benutzerdaten ein. *`MyCluster`*Ersetzen Sie es durch den Namen des Clusters, für den die Container-Instance registriert werden soll.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
EOF
```

Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

So aktivieren Sie den Spot-Instance-Ausgleich für eine vorhandene Container-Instance

1. Stellen Sie über SSH eine Verbindung mit der Spot-Instance her.
2. Bearbeiten Sie die Datei `/etc/ecs/ecs.config` und fügen Sie folgende Zeile hinzu:

   ```
   ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
   ```

3. Den Service `ecs` neu starten.

   - Für das Amazon-ECS-optimierte Amazon Linux 2-AMI:

     ```
     sudo systemctl restart ecs
     ```

4. (Optional) Durch Abfragen der Agenten-Introspektions-API-Operation können Sie überprüfen, ob der Agent ausgeführt wird und Sie können Informationen über Ihre neue Container-Instance einholen. Weitere Informationen finden Sie unter the section called “Introspektion von Containern”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Ausführen eines Skripts beim Starten einer Amazon ECS Linux-Container-Instance

Möglicherweise müssen Sie auf jeder Container-Instance einen bestimmten Container ausführen, um Betriebs- oder Sicherheitsprobleme wie Überwachung, Sicherheit, Metriken, Serviceerkennung oder Protokollierung zu lösen.

Zu diesem Zweck können Sie Ihre Container-Instances so konfigurieren, dass der Befehl docker run mit dem Benutzerdatenskript beim Start oder in manchen Init-Systemen wie Upstart oder systemd aufgerufen wird. Diese Methode funktioniert zwar, hat aber einige Nachteile, da Amazon ECS nichts über den Container weiß und CPU, Arbeitsspeicher, Ports oder sonstige verwendete Ressourcen nicht überwachen kann. Damit Amazon ECS alle Aufgabenressourcen ordnungsgemäß berücksichtigen kann, sollten Sie eine Aufgabendefinition für den Container erstellen, der auf Ihren Container-Instances ausgeführt werden soll. Verwenden Sie anschließend Amazon ECS, um die Aufgabe beim Start mit Amazon EC2-Benutzerdaten zu platzieren.

Bei dem Amazon EC2-Benutzerdatenskript im folgenden Verfahren wird die Amazon-ECS-Introspektions-API zur Ermittlung der Container-Instance verwendet. Anschließend verwendet es den Befehl AWS CLI und den start-task Befehl, um beim Start eine bestimmte Aufgabe für sich selbst auszuführen.

So starten Sie eine Aufgabe beim Start einer Container-Instance

1. Ändern Sie Ihre `ecsInstanceRole`-IAM-Rolle, um Berechtigungen für die API-Operation `StartTask` hinzuzufügen. Weitere Informationen finden Sie unter Ändern einer Rolle im AWS Identity and Access Management IAM-Benutzerhandbuch.
2. Starten Sie eine oder mehrere Container-Instances mit dem Amazon ECS-optimierten Amazon Linux 2-AMI. Starten Sie neue Container-Instances und verwenden Sie das folgende Beispielskript in den EC2-Benutzerdaten. Ersetzen *`Sie your_cluster_name`* durch den Cluster, in dem sich die Container-Instance registrieren soll, und *`my_task_def`* durch die Aufgabendefinition, die beim Start auf der Instance ausgeführt werden soll.

Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

> **Note**
>
> Der mehrteilige MIME-Inhalt unten verwendet zum Einstellen von Konfigurationswerten und Installieren von Paketen ein Shell-Skript. Außerdem verwendet er einen systemd-Auftrag zum Starten der Aufgabe, wenn der ecs-Service ausgeführt wird und die Introspektions-API verfügbar ist.

```
Content-Type: multipart/mixed; boundary="==BOUNDARY=="
MIME-Version: 1.0

--==BOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
# Specify the cluster that the container instance should register into
cluster=your_cluster_name

# Write the cluster configuration variable to the ecs.config file
# (add any other configuration variables here also)
echo ECS_CLUSTER=$cluster >> /etc/ecs/ecs.config

START_TASK_SCRIPT_FILE="/etc/ecs/ecs-start-task.sh"
cat <<- 'EOF' > ${START_TASK_SCRIPT_FILE}
 exec 2>>/var/log/ecs/ecs-start-task.log
 set -x

 # Install prerequisite tools
 yum install -y jq aws-cli

 # Wait for the ECS service to be responsive
 until curl -s http://localhost:51678/v1/metadata
 do
  sleep 1
 done

 # Grab the container instance ARN and AWS Region from instance metadata
 instance_arn=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F/ '{print $NF}' )
```

```
 cluster=$(curl -s http://localhost:51678/v1/metadata | jq -r '. | .Cluster' | awk
 -F/ '{print $NF}' )
 region=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F: '{print $4}')

 # Specify the task definition to run at launch
 task_definition=my_task_def

 # Run the AWS CLI start-task command to start your task on this container instance
 aws ecs start-task --cluster $cluster --task-definition $task_definition --
container-instances $instance_arn --started-by $instance_arn --region $region
EOF

# Write systemd unit file
UNIT="ecs-start-task.service"
cat <<- EOF > /etc/systemd/system/${UNIT}
      [Unit]
      Description=ECS Start Task
      Requires=ecs.service
      After=ecs.service

      [Service]
      Restart=on-failure
      RestartSec=30
      ExecStart=/usr/bin/bash ${START_TASK_SCRIPT_FILE}

      [Install]
      WantedBy=default.target
EOF

# Enable our ecs.service dependent service with `--no-block` to prevent systemd
 deadlock
# See https://github.com/aws/amazon-ecs-agent/issues/1707
systemctl enable --now --no-block "${UNIT}"
--==BOUNDARY==--
```

3. Überprüfen Sie, ob Ihre Container-Instances im richtigen Cluster gestartet werden und ob Ihre Aufgaben gestartet wurden.

   a. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.

   b. Wählen Sie auf der Navigationsleiste die Region aus, in der sich der Cluster befindet.

   c. Wählen Sie im Navigationsbereich Clusters und dann den Cluster aus, der Ihre Container-Instances hostet.

d. Wählen Sie auf der Seite Cluster Aufgaben und dann Ihre Aufgaben aus.

   Auf jeder Container-Instance, die Sie gestartet haben, sollte Ihre Aufgabe ausgeführt werden.

   Wenn Ihre Aufgaben nicht angezeigt werden, können Sie sich mit SSH bei Ihren Container-Instances anmelden und die Datei `/var/log/ecs/ecs-start-task.log` auf Debugging-Informationen überprüfen.

Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances

> **Note**
>
> Dieses Feature ist nicht auf Fargate verfügbar.

Jede Amazon ECS-Aufgabe, die den `awsvpc` Netzwerkmodus verwendet, erhält ihre eigene elastic network interface (ENI), die an die Container-Instance angehängt ist, die sie hostet. Es gibt eine Standardbegrenzung bei der Anzahl der Netzwerkschnittstellen, die an eine Amazon-EC2-Instance angehängt werden können. Dabei zählt die primäre Netzwerkschnittstelle als eine Einheit. Beispiel: Standardmäßig können an eine `c5.large`-Instance bis zu drei ENIs angehängt werden. Die primäre Netzwerkschnittstelle für die Instance zählt dazu, sodass sie zwei weitere ENIs anhängen können. Da für jede Aufgabe, die den `awsvpc` Netzwerkmodus verwendetENI, eine erforderlich ist, können Sie in der Regel nur zwei solcher Aufgaben auf diesem Instance-Typ ausführen.

Amazon ECS unterstützt das Starten von Container-Instances mit erhöhter ENI Dichte unter Verwendung unterstützter Amazon EC2 EC2-Instance-Typen. Wenn Sie diese Instance-Typen verwenden und die `awsvpcTrunking` Kontoeinstellungen aktivieren, sind zusätzliche ENIs für neu gestartete Container-Instances verfügbar. Diese Konfiguration gibt Ihnen die Möglichkeit, mehrere Aufgaben auf den einzelnen Container-Instances zu platzieren. Informationen zur `awsvpcTrunking` Kontoeinstellung finden Sie unterGreifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu.

Beispiel: Eine `c5.large` Instanz mit `awsvpcTrunking` hat ein erhöhtes ENI Limit von zwölf. Die Container-Instance verfügt über die primäre Netzwerkschnittstelle und Amazon ECS erstellt und fügt eine "Stamm"-Netzwerkschnittstelle an die Container-Instance an. Mit dieser Konfiguration können Sie also zehn Aufgaben anstelle der aktuellen zwei Aufgaben auf der Container-Instance starten.

Die Stamm-Netzwerkschnittstelle wird vollständig von Amazon ECS verwaltet. Sie wird gelöscht, wenn Sie Ihre Container-Instance beenden oder die Registrierung auf dem Cluster aufheben. Weitere Informationen finden Sie unter Netzwerkoptionen für Amazon ECS-Aufgaben für den EC2-Starttyp.

Überlegungen

Beachten Sie Folgendes, wenn Sie die ENI Trunking-Funktion verwenden.

- Nur Linux-Varianten des Amazon ECS-optimierten AMI oder andere Amazon Linux-Varianten mit Version `1.28.1` oder höher des Container-Agents und Version `1.28.1-2` oder höher des ecs-init-Pakets unterstützen die erhöhten Grenzwerte. ENI Wenn Sie die neueste Linux-Variante des Amazon-ECS-optimierten AMI verwenden, sind diese Anforderungen erfüllt. Windows-Container werden derzeit nicht unterstützt.
- Nur neue Amazon EC2 EC2-Instances, die nach der Aktivierung gestartet werden, `awsvpcTrunking` erhalten die erhöhten ENI Grenzwerte und die Trunk-Netzwerkschnittstelle. Zuvor gestarteten Instances erhalten keine dieser Features, unabhängig von den durchgeführten Aktionen.
- Amazon-EC2-Instances müssen ressourcenbasierte IPv4-DNS-Anforderungen deaktiviert haben. Stellen Sie zum Deaktivieren dieser Option sicher, dass die Option Enable resource-based IPV4 (A record) DNS requests (Ressourcenbasierte IPV4-DNS-Anforderungen (A-Eintrag) aktivieren) deaktiviert ist, wenn Sie eine neue Instance mit der Amazon-EC2-Konsole erstellen. Verwenden Sie den folgenden Befehl AWS CLI, um diese Option mit dem zu deaktivieren.

  ```
  aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-name-dns-a-record --no-dry-run
  ```

- Amazon-EC2-Instances in freigegebenen Subnetzen werden nicht unterstützt. Sie können sich nicht bei einem Cluster registrieren, wenn sie verwendet werden.
- Ihre Amazon-ECS-Aufgaben müssen den Netzwerkmodus `awsvpc` und den Starttyp EC2 verwenden. Aufgaben, die den Starttyp Fargate verwenden, erhielten immer einen dedizierten ENI Status, unabhängig davon, wie viele gestartet wurden, sodass diese Funktion nicht benötigt wird.
- Ihre Amazon ECS Aufgaben müssen in derselben Amazon VPC wie Ihre Container-Instance gestartet werden. Beim Starten Ihrer Aufgaben kommt es zu einem Attributfehler, wenn sich die Aufgaben nicht in derselben VPC befinden.
- Beim Starten einer neuen Container-Instance wechselt die Instance in den Status `REGISTERING` und die Stamm-ENI wird für die Instance bereitgestellt. Wenn die Registrierung fehlschlägt, wechselt die Instance in den Status `REGISTRATION_FAILED`. Sie können Probleme bei einer

fehlgeschlagenen Registrierung beheben, indem Sie die Container-Instance beschreiben, um das Feld `statusReason` einzusehen, welches den Grund für den Fehler beschreibt. Die Container-Instance kann dann manuell abgemeldet oder beendet werden. Sobald die Container-Instance erfolgreich deregistriert oder beendet wurde, löscht Amazon ECS den Trunk. ENI

> **Note**
>
> Amazon ECS sendet Statusänderungsereignisse für Container-Instances aus, die Sie für Instances überwachen können, die zu einem `REGISTRATION_FAILED`-Zustand übergehen. Weitere Informationen finden Sie unter Ereignisse zur Änderung des Status der Amazon ECS-Container-Instance.

- Sobald die Container-Instance beendet wird, wechselt die Instance in den Status `DEREGISTERING` und die Bereitstellung der Stamm-ENI wird aufgehoben. Die Instance wechselt dann in einen `INACTIVE`-Status.
- Wenn eine Container-Instance in einem öffentlichen Subnetz mit den erhöhten ENI Grenzwerten gestoppt und anschließend neu gestartet wird, verliert die Instance ihre öffentliche IP-Adresse und der Container-Agent verliert seine Verbindung.
- Wenn Sie diese Option aktivieren`awsvpcTrunking`, erhalten Container-Instances eine zusätzlicheENI, die die Standardsicherheitsgruppe der VPC verwendet und von Amazon ECS verwaltet wird.

Voraussetzungen

Bevor Sie eine Container-Instance mit den erhöhten ENI Limits starten, müssen die folgenden Voraussetzungen erfüllt sein.

- Die serviceverknüpfte Rolle für Amazon ECS muss erstellt werden. Die serviceverknüpfte Amazon ECS-Rolle gibt Amazon ECS die Erlaubnis, in Ihrem Namen Anrufe an andere AWS Services zu tätigen. Diese Rolle wird automatisch für Sie erstellt, wenn Sie einen Cluster erstellen oder wenn Sie einen Service in der AWS Management Console erstellen oder aktualisieren. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS. Sie können die serviceverknüpfte Rolle auch mit dem folgenden AWS CLI Befehl erstellen.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Die IAM-Rolle Ihres Kontos oder Ihrer Container-Instance muss die `awsvpcTrunking`-Kontoeinstellung aktivieren. Wir empfehlen, dass Sie 2 Container-Instance-Rollen (`ecsInstanceRole`) erstellen. Sie können dann die `awsvpcTrunking` Kontoeinstellung für eine Rolle aktivieren und diese Rolle für Aufgaben verwenden, die ENI-Trunking erfordern. Informationen zur Container-Instance-Rolle finden Sie unterIAM-Rolle für Amazon-ECS-Container-Instance.

Sobald die Voraussetzungen erfüllt sind, können Sie eine neue Container-Instance mit einem der unterstützten Amazon EC2 EC2-Instance-Typen starten. Für die Instance gelten dann die erhöhten ENI Limits. Eine Liste mit unterstützten Instance-Typen finden Sie unter Unterstützte Instances für mehr Amazon ECS-Container-Netzwerkschnittstellen. Die Container-Instance muss Version `1.28.1` oder höher des Container-Agenten haben sowie Version `1.28.1-2` oder höher des ecs-Init-Pakets. Wenn Sie die neueste Linux-Variante des Amazon-ECS-optimierten AMI verwenden, sind diese Anforderungen erfüllt. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

> **Important**
>
> Amazon-EC2-Instances müssen ressourcenbasierte IPv4-DNS-Anforderungen deaktiviert haben. Stellen Sie zum Deaktivieren dieser Option sicher, dass die Option Enable resource-based IPV4 (A record) DNS requests (Ressourcenbasierte IPV4-DNS-Anforderungen (A-Eintrag) aktivieren) deaktiviert ist, wenn Sie eine neue Instance mit der Amazon-EC2-Konsole erstellen. Verwenden Sie den folgenden Befehl AWS CLI, um diese Option mit dem zu deaktivieren.
>
> ```
> aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-name-dns-a-record --no-dry-run
> ```

Um Ihre Container-Instances mit erhöhten ENI Limits anzuzeigen, verwenden Sie den AWS CLI

Jede Container-Instance verfügt über eine standardmäßige Netzwerkschnittstelle, die als Stamm-Netzwerkschnittstelle bezeichnet wird. Verwenden Sie den folgenden Befehl, um Ihre Container-Instances mit erhöhten ENI Limits aufzulisten, indem Sie nach dem `ecs.awsvpc-trunk-id` Attribut fragen, das angibt, dass sie über eine Trunk-Netzwerkschnittstelle verfügen.

- list-attributes (AWS CLI)

```
aws ecs list-attributes \
      --target-type container-instance \
      --attribute-name ecs.awsvpc-trunk-id \
      --cluster cluster_name \
      --region us-east-1
```

- Get-ECS AttributeList ()AWS Tools for Windows PowerShell

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-
id -Region us-east-1
```

Unterstützte Instances für mehr Amazon ECS-Container-Netzwerkschnittstellen

Nachfolgend finden Sie die unterstützten Amazon-EC2-Instance-Typen und die Anzahl der Aufgaben, die auf dem Instance-Typ mit dem Netzwerkmodus `awsvpc` gestartet werden können und zwar vor und nach dem Aktivieren der `awsvpcTrunking`-Kontoeinstellung. Fügen Sie für die elastic network interface (ENI) -Limits für jeden Instance-Typ eins zum aktuellen Task-Limit hinzu, da die primäre Netzwerkschnittstelle auf das Limit angerechnet wird, und fügen Sie zwei zum neuen Task-Limit hinzu, da sowohl die primäre Netzwerkschnittstelle als auch die Trunk-Netzwerkschnittstelle das Limit erneut zählen.

> **Important**
>
> Obwohl andere Instance-Typen in derselben Instance-Familie unterstützt werden, werden die Instance-Typen `a1.metal`, `c5.metal`, `c5a.8xlarge`, `c5ad.8xlarge`, `c5d.metal`, `m5.metal`, `p3dn.24xlarge`, `r5.metal`, `r5.8xlarge` und `r5d.metal` nicht unterstützt. Die Instance-Familien `c5n`, `d3`, `d3en`, `g3`, `g3s`, `g4dn`, `i3`, `i3en`, `inf1`, `m5dn`, `m5n`, `m5zn`, `mac1`, `r5b`, `r5n`, `r5dn`, `u-12tb1`, `u-6tb1`, `u-9tb1` und `z1d` werden nicht unterstützt.

Themen

- Allgemeine Zwecke
- Für Datenverarbeitung optimiert
- RAM-optimiert
- Speicheroptimiert
- Beschleunigtes Computing

- High Performance Computing

Allgemeine Zwecke

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
a1.medium	1	10
a1.large	2	10
a1.xlarge	3	20
a1.2xlarge	3	40
a1.4xlarge	7	60
m5.large	2	10
m5.xlarge	3	20
m5.2xlarge	3	40
m5.4xlarge	7	60
m5.8xlarge	7	60
m5.12xlarge	7	60
m5.16xlarge	14	120
m5.24xlarge	14	120
m5a.large	2	10
m5a.xlarge	3	20
m5a.2xlarge	3	40
m5a.4xlarge	7	60

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m5a.8xlarge	7	60
m5a.12xlarge	7	60
m5a.16xlarge	14	120
m5a.24xlarge	14	120
m5ad.large	2	10
m5ad.xlarge	3	20
m5ad.2xlarge	3	40
m5ad.4xlarge	7	60
m5ad.8xlarge	7	60
m5ad.12xlarge	7	60
m5ad.16xlarge	14	120
m5ad.24xlarge	14	120
m5d.large	2	10
m5d.xlarge	3	20
m5d.2xlarge	3	40
m5d.4xlarge	7	60
m5d.8xlarge	7	60
m5d.12xlarge	7	60
m5d.16xlarge	14	120
m5d.24xlarge	14	120

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m5d.metal	14	120
m5n.large	2	10
m5n.xlarge	3	20
m5n.2xlarge	3	40
m5n.4xlarge	7	60
m5n.8xlarge	7	60
m5n.12xlarge	7	60
m5n.16xlarge	14	120
m5zn.large	2	14
m5zn.xlarge	3	31
m5zn.2xlarge	3	64
m5zn.3xlarge	7	98
m5zn.6xlarge	7	120
m6a.large	2	10
m6a.xlarge	3	20
m6a.2xlarge	3	40
m6a.4xlarge	7	60
m6a.8xlarge	7	90
m6a.12xlarge	7	120
m6a.16xlarge	14	120

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m6a.24xlarge	14	120
m6a.32xlarge	14	120
m6a.48xlarge	14	120
m6a.metal	14	120
m6g.medium	1	4
m6g.large	2	10
m6g.xlarge	3	20
m6g.2xlarge	3	40
m6g.4xlarge	7	60
m6g.8xlarge	7	60
m6g.12xlarge	7	60
m6g.16xlarge	14	120
m6g.metal	14	120
m6gd.medium	1	4
m6gd.large	2	10
m6gd.xlarge	3	20
m6gd.2xlarge	3	40
m6gd.4xlarge	7	60
m6gd.8xlarge	7	60
m6gd.12xlarge	7	60

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m6gd.16xlarge	14	120
m6gd.metal	14	120
m6i.large	2	10
m6i.xlarge	3	20
m6i.2xlarge	3	40
m6i.4xlarge	7	60
m6i.8xlarge	7	90
m6i.12xlarge	7	120
m6i.16xlarge	14	120
m6i.24xlarge	14	120
m6i.32xlarge	14	120
m6i.metal	14	120
m6id.large	2	10
m6id.x groß	3	20
m6id.2xlarge	3	40
m6id.4xlarge	7	60
m6id.8xlarge	7	90
m6id.12xlarge	7	120
m6id.16xlarge	14	120
m6id.24xlarge	14	120

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m6id.32xlarge	14	120
m6id.metal	14	120
m6idn.large	2	10
m6idn.xlarge	3	20
m6id.2xlarge	3	40
m6idn.4xlarge	7	60
m6idn.8xlarge	7	90
m6idn.12xlarge	7	120
m6idn.16xlarge	14	120
m6idn.24xlarge	14	120
m6idn.32xlarge	15	120
m6idn.metal	15	120
m6in.large	2	10
m6in.xlarge	3	20
m6in.2xlarge	3	40
m6in.4xlarge	7	60
m6n.8xlarge	7	90
m6in.12xlarge	7	120
m6in.16xlarge	14	120
m6in.24xlarge	14	120

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m6in.32xlarge	15	120
m6in.metal	15	120
m7a.medium	1	4
m7a.large	2	10
m7a.xlarge	3	20
m7a.2xlarge	3	40
m7a.4xlarge	7	60
m7a.8xlarge	7	90
m7a.12xlarge	7	120
m7a.16xlarge	14	120
m7a.24xlarge	14	120
m7a.32xlarge	14	120
m7a.48xlarge	14	120
m7a.metal-48xl	14	120
m7g.medium	1	4
m7g.large	2	10
m7g.xlarge	3	20
m7g.2xlarge	3	40
m7g.4xlarge	7	60
m7g.8xlarge	7	60

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m7g.12xlarge	7	60
m7g.16xlarge	14	120
m7g.metal	14	120
m7gd.medium	1	4
m7gd.large	2	10
m7gd.xlarge	3	20
m7gd.2xlarge	3	40
m7gd.4xlarge	7	60
m7gd.8xlarge	7	60
m7gd.12xlarge	7	60
m7gd.16xlarge	14	120
7 mg. Metall	14	120
m7i.large	2	10
m7i.xlarge	3	20
m7i.2xlarge	3	40
m7i.4xlarge	7	60
m7i.8xlarge	7	90
m7i.12xlarge	7	120
m7i.16xlarge	14	120
m7i.24xlarge	14	120

Instance-Typ	Aufgabenlimit ohne ENI Trunking	Aufgabenlimit mit ENI Bündelung
m7i.48xlarge	14	120
m7i.metal-24xl	14	120
m7i.metal-48xl	14	120
m7i-flex.large	2	4
m7i-flex.xlarge	3	10
m7i-flex.2xlarge	3	20
m7i-flex.4xlarge	7	40
m7i-flex.8xlarge	7	60
mac2.metal	7	12
mac2-m2.metal	7	12
mac2-m2pro.metal	7	12

Für Datenverarbeitung optimiert

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c5.large	2	10
c5.xlarge	3	20
c5.2xlarge	3	40
c5.4xlarge	7	60
c5.9xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c5.12xlarge	7	60
c5.18xlarge	14	120
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20
c5a.2xlarge	3	40
c5a.4xlarge	7	60
c5a.12xlarge	7	60
c5a.16xlarge	14	120
c5a.24xlarge	14	120
c5ad.large	2	10
c5ad.xlarge	3	20
c5ad.2xlarge	3	40
c5ad.4xlarge	7	60
c5ad.12xlarge	7	60
c5ad.16xlarge	14	120
c5ad.24xlarge	14	120
c5d.large	2	10
c5d.xlarge	3	20
c5d.2xlarge	3	40

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c5d.4xlarge	7	60
c5d.9xlarge	7	60
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120
c6a.large	2	10
c6a.xlarge	3	20
c6a.2xlarge	3	40
c6a.4xlarge	7	60
c6a.8xlarge	7	90
c6a.12xlarge	7	120
c6a.16xlarge	14	120
c6a.24xlarge	14	120
c6a.32xlarge	14	120
c6a.48xlarge	14	120
c6a.metal	14	120
c6g.medium	1	4
c6g.large	2	10
c6g.xlarge	3	20
c6g.2xlarge	3	40

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c6g.4xlarge	7	60
c6g.8xlarge	7	60
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120
c6gd.medium	1	4
c6gd.large	2	10
c6gd.xlarge	3	20
c6gd.2xlarge	3	40
c6gd.4xlarge	7	60
c6gd.8xlarge	7	60
c6gd.12xlarge	7	60
c6gd.16xlarge	14	120
c6gd.metal	14	120
c6gn.medium	1	4
c6gn.large	2	10
c6gn.xlarge	3	20
c6gn.2xlarge	3	40
c6gn.4xlarge	7	60
c6gn.8xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c6gn.12xlarge	7	60
c6gn.16xlarge	14	120
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40
c6i.4xlarge	7	60
c6i.8xlarge	7	90
c6i.12xlarge	7	120
c6i.16xlarge	14	120
c6i.24xlarge	14	120
c6i.32xlarge	14	120
c6i.metal	14	120
c6id.large	2	10
c6id.xlarge	3	20
c6id.2xlarge	3	40
c6id.4xlarge	7	60
c6id.8xlarge	7	90
c6id.12xlarge	7	120
c6id.16xlarge	14	120
c6id.24xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c6id.32xlarge	14	120
c6id.metal	14	120
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	15	120
c6in.metal	15	120
c7a.medium	1	4
c7a.large	2	10
c7a.xlarge	3	20
c7a.2xlarge	3	40
c7a.4xlarge	7	60
c7a.8xlarge	7	90
c7a.12xlarge	7	120
c7a.16xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c7a.24xlarge	14	120
c7a.32xlarge	14	120
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4
c7g.large	2	10
c7g.xlarge	3	20
c7g.2xlarge	3	40
c7g.4xlarge	7	60
c7g.8xlarge	7	60
c7g.12xlarge	7	60
c7g.16xlarge	14	120
c7g.metal	14	120
c7gd.medium	1	4
c7gd.large	2	10
c7gd.xlarge	3	20
c7gd.2xlarge	3	40
c7gd.4xlarge	7	60
c7gd.8xlarge	7	60
c7gd.12xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c7gd.16xlarge	14	120
c7gd.metal	14	120
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	14	120
C7Gn. Metall	14	120
c7i. groß	2	10
c7i.x groß	3	20
c7i.2xlarge	3	40
c7i.4xlarge	7	60
c7i.8xlarge	7	90
c7i.12xlarge	7	120
c7i.16xlarge	14	120
c7i.24xlarge	14	120
c7i.48xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
c7i.metal-24xl	14	120
c7i.metal-48xl	14	120
C7i-Flex. Groß	2	4
c7i-flex.xgroß	3	10
c7i-flex.2 x groß	3	20
c7i-flex.4x groß	7	40
c7i-flex.8 x groß	7	60

RAM-optimiert

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r5.large	2	10
r5.xlarge	3	20
r5.2xlarge	3	40
r5.4xlarge	7	60
r5.12xlarge	7	60
r5.16xlarge	14	120
r5.24xlarge	14	120
r5a.large	2	10
r5a.xlarge	3	20

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r5a.2xlarge	3	40
r5a.4xlarge	7	60
r5a.8xlarge	7	60
r5a.12xlarge	7	60
r5a.16xlarge	14	120
r5a.24xlarge	14	120
r5ad.large	2	10
r5ad.xlarge	3	20
r5ad.2xlarge	3	40
r5ad.4xlarge	7	60
r5ad.8xlarge	7	60
r5ad.12xlarge	7	60
r5ad.16xlarge	14	120
r5ad.24xlarge	14	120
r5b.16xlarge	14	120
r5d.large	2	10
r5d.xlarge	3	20
r5d.2xlarge	3	40
r5d.4xlarge	7	60
r5d.8xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r5d.12xlarge	7	60
r5d.16xlarge	14	120
r5d.24xlarge	14	120
r5dn.16xlarge	14	120
r6a.large	2	10
r6a.xlarge	3	20
r6a.2xlarge	3	40
r6a.4xlarge	7	60
r6a.8xlarge	7	90
r6a.12xlarge	7	120
r6a.16xlarge	14	120
r6a.24xlarge	14	120
r6a.32xlarge	14	120
r6a.48xlarge	14	120
r6a.metal	14	120
r6g.medium	1	4
r6g.groß	2	10
r6g.xlarge	3	20
r6g.2xlarge	3	40
r6g.4xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r6g.8xlarge	7	60
r6g.12xlarge	7	60
r6g.16xlarge	14	120
r6g.metal	14	120
r6gd.medium	1	4
r6gd.large	2	10
r6gd.xlarge	3	20
r6gd.2xlarge	3	40
r6gd.4xlarge	7	60
r6gd.8xlarge	7	60
r6gd.12xlarge	7	60
r6gd.16xlarge	14	120
r6gd.metal	14	120
r6i.large	2	10
r6i.xlarge	3	20
r6i.2xlarge	3	40
r6i.4xlarge	7	60
r6i.8xlarge	7	90
r6i.12xlarge	7	120
r6i.16xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r6i.24xlarge	14	120
r6i.32xlarge	14	120
r6i.metal	14	120
r6idn.large	2	10
r6idn.xlarge	3	20
r6idn.2xlarge	3	40
r6idn.4xlarge	7	60
r6idn.8xlarge	7	90
r6idn.12xlarge	7	120
r6idn.16xlarge	14	120
r6idn.24xlarge	14	120
r6idn.32xlarge	15	120
r6idn.metal	15	120
r6in.large	2	10
r6in.xlarge	3	20
r6in.2xlarge	3	40
r6in.4xlarge	7	60
r6in.8xlarge	7	90
r6in.12xlarge	7	120
r6in.16xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r6in.24xlarge	14	120
r6in.32xlarge	15	120
r6in.metal	15	120
r6id.large	2	10
r6id.xlarge	3	20
r6id.2xlarge	3	40
r6id.4xlarge	7	60
r6id.8xlarge	7	90
r6id.12xlarge	7	120
r6id.16xlarge	14	120
r6id.24xlarge	14	120
r6id.32xlarge	14	120
r6id.metal	14	120
r7a.medium	1	4
r7a.large	2	10
r7a.x groß	3	20
r7a.2xlarge	3	40
r7a.4xlarge	7	60
r7a.8xlarge	7	90
r7a.12xlarge	7	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r7a.16xlarge	14	120
r7a.24xlarge	14	120
r7a.32xlarge	14	120
r7a.48xlarge	14	120
r7a.metal-48xl	14	120
r7g.medium	1	4
r7g.large	2	10
r7g.xlarge	3	20
r7g.2xlarge	3	40
r7g.4xlarge	7	60
r7g.8xlarge	7	60
r7g.12xlarge	7	60
r7g.16xlarge	14	120
r7g.metal	14	120
r7gd.medium	1	4
r7gd.large	2	10
r7gd.xlarge	3	20
r7gd.2xlarge	3	40
r7gd.4xlarge	7	60
r7gd.8xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r7gd.12xlarge	7	60
r7gd.16xlarge	14	120
r7gd.metal	14	120
r7i.large	2	10
r7i.xlarge	3	20
r7i.2xlarge	3	40
r7i.4xlarge	7	60
r7i.8xlarge	7	90
r7i.12xlarge	7	120
r7i.16xlarge	14	120
r7i.24xlarge	14	120
r7i.48xlarge	14	120
r7i.metal-24xl	14	120
r7i.metal-48xl	14	120
r7iz.large	2	10
r7iz.xlarge	3	20
r7iz.2xlarge	3	40
r7iz.4xlarge	7	60
r7iz.8xlarge	7	90
r7iz.12xlarge	7	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
r7iz.16xlarge	14	120
r7iz.32xlarge	14	120
r7iz.metal-16xl	14	120
r7iz.metal-32xl	14	120
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12
u7i-12tb.224x groß	14	120
u7in-16 tb.224x groß	15	120
u7in-24 tb.224x groß	15	120
u7in-32 TB. 224x groß	15	120
x2gd.medium	1	10
x2gd.large	2	10
x2gd.xlarge	3	20
x2gd.2xlarge	3	40
x2gd.4xlarge	7	60
x2gd.8xlarge	7	60

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
x2gd.12xlarge	7	60
x2gd.16xlarge	14	120
x2gd.metal	14	120
x2idn.16xlarge	14	120
x2idn.24xlarge	14	120
x2idn.32xlarge	14	120
x2idn.metal	14	120
x2iedn.xlarge	3	13
x2iedn.2xlarge	3	29
x2iedn.4xlarge	7	60
x2iedn.8xlarge	7	120
x2iedn.16xlarge	14	120
x2iedn.24xlarge	14	120
x2iedn.32xlarge	14	120
x2iedn.metal	14	120
x2iezn.2xlarge	3	64
x2iezn.4xlarge	7	120
x2iezn.6xlarge	7	120
x2iezn.8xlarge	7	120
x2iezn.12xlarge	14	120

Instance-Typ	Aufgabenlimit ohne Bündelung ENI	Aufgabenlimit mit ENI Bündelung
x2iezn.metal	14	120

Speicheroptimiert

Instance-Typ	Aufgabenlimit ohne Trunking ENI	Aufgabenlimit mit ENI Bündelung
i4g.large	2	10
i4g.xlarge	3	20
i4g.2xlarge	3	40
i4g.4xlarge	7	60
i4g.8xlarge	7	60
i4g.16xlarge	14	120
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498
i4i.metal	14	498

Instance-Typ	Aufgabenlimit ohne Trunking ENI	Aufgabenlimit mit ENI Bündelung
im4gn.large	2	10
im4gn.xlarge	3	20
im4gn.2xlarge	3	40
im4gn.4xlarge	7	60
im4gn.8xlarge	7	60
im4gn.16xlarge	14	120
is4gen.medium	1	4
is4gen.large	2	10
is4gen.xlarge	3	20
is4gen.2xlarge	3	40
is4gen.4xlarge	7	60
is4gen.8xlarge	7	60

Beschleunigtes Computing

Instance-Typ	Aufgabenlimit ohne Trunking ENI	Aufgabenlimit mit ENI Bündelung
dl1.24xlarge	59	120
dl2q.24xlarge	14	120
g4ad.xlarge	1	12
g4ad.2xlarge	1	12

Instance-Typ	Aufgabenlimit ohne Trunking ENI	Aufgabenlimit mit ENI Bündelung
g4ad.4xlarge	2	12
g4ad.8xlarge	3	12
g4ad.16xlarge	7	12
g5.xgroß	3	6
g5.2xgroß	3	19
g5.4xlarge	7	40
g5.8xlarge	7	90
g5.12xlarge	14	120
g5.16xlarge	7	120
g5.24xlarge	14	120
g5.48xlarge	6	120
g5g.xlarge	3	20
g5g.2xlarge	3	40
g5g.4xlarge	7	60
g5g.8xlarge	7	60
g5g.16xlarge	14	120
g5g.metal	14	120
g6.x groß	3	20
g 6.2 x groß	3	40
g 6,4 x groß	7	60

Instance-Typ	Aufgabenlimit ohne Trunking ENI	Aufgabenlimit mit ENI Bündelung
g 6,8 x groß	7	90
g 6.12 x groß	7	120
g 6.16 x groß	14	120
g 6.24 x groß	14	120
g 6,48 x groß	14	120
gr 6,4 x groß	7	60
gr 6,8 x groß	7	90
inf2.xlarge	3	20
inf2.8xlarge	7	90
inf2.24xlarge	14	120
inf2.48xlarge	14	120
p4d.24xgroß	59	120
p4de.24xlarge	59	120
p5.48xlarge	63	242
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

High Performance Computing

Instance-Typ	Aufgabenlimit ohne ENI Bündelung	Aufgabenlimit mit ENI Bündelung
hpc6a.48xlarge	1	120
hpc6id.32xlarge	1	120
hpc7g.4xlarge	3	120
hpc7g.8xlarge	3	120
hpc7g.16xlarge	3	120

Reservieren von Amazon ECS Linux-Container-Instance-Speicher

Wenn der Amazon ECS-Container-Agent eine Container-Instance in einem Cluster registriert, muss der Agent ermitteln, wie viel Speicher die Container-Instance für Ihre Aufgaben zur Verfügung hat. Aufgrund des Speicher-Overheads der Plattform und des vom System-Kernel belegten Arbeitsspeichers weicht dieser Wert vom Wert des installierten Arbeitsspeichers ab, der für Amazon-EC2-Instances angegeben wird. Eine `m4.large`-Instance beispielsweise besitzt 8 GiB installierten Speicher. Dies bedeutet jedoch nicht immer, dass genau 8192 MiB Speicher für Aufgaben zur Verfügung stehen, wenn sich die Container-Instance registriert.

Der Amazon ECS-Container-Agent stellt eine Konfigurationsvariable namens bereit`ECS_RESERVED_MEMORY`, mit der Sie eine bestimmte Anzahl von MiB Arbeitsspeicher aus dem Pool entfernen können, der Ihren Aufgaben zugewiesen ist. Damit wird dieser Arbeitsspeicher für wichtige Systemprozesse reserviert.

Wenn Sie den gesamten Speicher einer Container-Instance mit Ihren Aufgaben belegen, ist es möglich, dass Ihre Aufgaben mit kritischen Systemprozessen in Bezug auf den Arbeitsspeicher zu kämpfen haben und möglicherweise einen Systemausfall auslösen.

Wenn Sie beispielsweise `ECS_RESERVED_MEMORY=256` in der Konfigurationsdatei Ihres Container-Agenten angeben, registriert der Agent den gesamten Arbeitsspeicher bis auf 256 MiB für diese Instance, und 256 MiB Arbeitsspeicher können nicht für ECS-Aufgaben zugeordnet werden. Weitere Informationen über die Konfigurationsvariablen des Agenten und ihre Einstellung finden Sie unter

Konfiguration des Amazon-ECS-Container-Agenten und Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe.

Wenn Sie 8192 MiB für die Aufgabe angeben und keine Ihrer Container-Instances über 8192 MiB oder mehr Arbeitsspeicher verfügt, um diese Anforderung zu erfüllen, kann die Aufgabe nicht in Ihrem Cluster platziert werden. Wenn Sie eine verwaltete Rechenumgebung verwenden, AWS Batch müssen Sie einen größeren Instance-Typ starten, um der Anfrage gerecht zu werden.

Sie sollten außerdem etwas Arbeitsspeicher für den Amazon-ECS-Container-Agenten und andere wichtige Systemprozesse in Ihren Container-Instances reservieren, sodass die Container für Ihre Aufgaben nicht um denselben Arbeitsspeicher konkurrieren und möglicherweise einen Systemfehler auslösen.

Der Amazon-ECS-Container-Agent verwendet die Docker-Funktion `ReadMemInfo()` für die Abfrage des gesamten für das Betriebssystem verfügbaren Arbeitsspeichers. Sowohl Linux als auch Windows stellen Befehlszeilenprogramme zur Verfügung, mit denen der Gesamtspeicher ermittelt werden kann.

Example – Bestimmung des Gesamtspeichers für Linux

Der Befehl free gibt Informationen darüber zurück, wie viel Speicher insgesamt vom Betriebssystem erkannt wird.

```
$ free -b
```

Beispielausgabe für eine `m4.large`-Instance, die das Amazon-ECS-optimierte Amazon Linux-AMI ausführt.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Diese Instance verfügt über 8373026816 Bytes Gesamtspeicher, und es stehen 7985 MiB für Aufgaben zur Verfügung.

Example – Bestimmung des Gesamtspeichers für Windows

Der Befehl wmic gibt Informationen darüber zurück, wie viel Speicher insgesamt vom Betriebssystem erkannt wird.

```
C:\> wmic ComputerSystem get TotalPhysicalMemory
```

Beispielausgabe für eine `m4.large` Instance, auf der das Amazon ECS-optimierte Windows Server-AMI ausgeführt wird.

```
TotalPhysicalMemory
8589524992
```

Diese Instance verfügt über 8589524992 Bytes Gesamtspeicher, und es stehen 8191 MiB für Aufgaben zur Verfügung.

Speicher der Container-Instance anzeigen

Sie können in der Amazon ECS-Konsole (oder mit dem Instance-API-Vorgang) sehen, mit wie viel Speicher sich eine DescribeContainerContainer-Instance registriert. Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Aufgaben so viel Speicher wie möglich für einen bestimmten Instance-Typ zur Verfügung stellen, können Sie den für diese Container-Instance verfügbaren Speicher beobachten und Ihren Aufgaben dann so viel Speicher zuweisen.

Um den Speicher der Container-Instance anzuzeigen

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Clusters und dann den Cluster aus, der Ihre Container-Instance hostet.
3. Wählen Sie Infrastruktur und dann unter Container-Instances eine Container-Instance aus.
4. Im Abschnitt Ressourcen wird der registrierte und verfügbare Speicher für die Container-Instance angezeigt.

   Der Wert Registered Memory entspricht dem, was die Container-Instance bei Amazon ECS registriert hat, als sie zum ersten Mal gestartet wurde, und der Wert Available memory ist der Wert, der noch nicht Aufgaben zugewiesen wurde.

Fernverwaltung von Amazon ECS-Container-Instances mithilfe von AWS Systems Manager

Sie können die Funktion Run Command in AWS Systems Manager (Systems Manager) verwenden, um die Konfiguration Ihrer Amazon ECS-Container-Instances sicher und remote zu verwalten. Run Command bietet eine einfache Möglichkeit, allgemeine Verwaltungsaufgaben ohne lokale Anmeldung

bei der Instance auszuführen. Sie können Konfigurationsänderungen für all Ihre Cluster verwalten, indem Sie Befehle für mehrere Container-Instances gleichzeitig ausführen. Run Command erstellt Berichte über den Status und die Ergebnisse jedes Befehls.

Hier finden Sie einige Beispiele der Arten von Aufgaben, die Sie mit Run Command ausführen können:

- Installieren oder deinstallieren von Paketen.
- Ausführen von Sicherheitsupdates.
- Bereinigen von Docker-Images.
- Beenden oder Starten von Services.
- Anzeigen von Systemressourcen.
- Anzeigen von Protokolldateien.
- Durchführen von Dateioperationen.

Weitere Informationen zu Run Command finden Sie unter AWS Systems Manager -Run Command im AWS Systems Manager -Benutzerhandbuch.

Im Folgenden sind die Voraussetzungen für die Verwendung von Systems Manager mit Amazon ECS aufgeführt.

1. Sie müssen der Container-Instance-Rolle (ecs InstanceRole) Berechtigungen für den Zugriff auf die Systems Manager Manager-APIs gewähren. Sie können dies tun, indem Sie der Rolle den AmazonSSM ManagedInstance Core zuweisen. `ecsInstanceRole` Informationen zum Anhängen einer Richtlinie an eine Rolle finden Sie unter Ändern einer Rollenberechtigungsrichtlinie (Konsole) im Benutzerhandbuch AWS Identity and Access Management
2. Stellen Sie sicher, dass SSM Agent auf Ihrer Container-Instance installiert ist. Weitere Informationen finden Sie unter Manuelles Installieren eines SSM Agents auf EC2-Instances für Linux.

Nachdem Sie Ihre von Systems Manager verwalteten Richtlinien hinzugefügt haben `ecsInstanceRole` und sich vergewissert haben, dass AWS Systems Manager Agent (SSM Agent) auf Ihren Container-Instances installiert ist, können Sie mit Run Command Befehle an Ihre Container-Instances senden. Weitere Informationen zum Ausführen von Befehlen und Shell-Scripts auf Ihren Instances sowie zum Anzeigen der Ausgabe finden Sie unter Ausführen von Befehlen mit

Systems Manager Run Command und Run Command Walkthroughs im AWS Systems Manager - Benutzerhandbuch.

Ein häufiger Anwendungsfall ist die Aktualisierung der Container-Instance-Software mit Run Command. Sie können den Verfahren im AWS Systems Manager Benutzerhandbuch mit den folgenden Parametern folgen.

Parameter	Wert
Befehlsdokument	`AWS-RunShellScript`
Befehl	`$ yum update -y`
Zielinstanzen	Ihre Container-Instances

Verwenden eines HTTP-Proxys für Amazon ECS Linux-Container-Instances

Sie können Ihre Amazon-ECS-Container-Instances für die Verwendung eines HTTP-Proxys sowohl für den Amazon-ECS-Container-Agenten als auch für den Docker-Daemon konfigurieren. Das ist praktisch, wenn Ihre Container-Instances keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance haben.

Um Ihre Amazon-ECS-Linux-Container-Instance für die Verwendung eines HTTP-Proxys zu konfigurieren, bestimmen Sie beim Starten die folgenden Variablen in den relevanten Dateien (mit Amazon EC2-Benutzerdaten). Sie können die Konfigurationsdatei auch manuell bearbeiten und dann den Agenten neu starten.

`/etc/ecs/ecs.config`(Amazon Linux 2und AmazonLinux AMI)

`HTTP_PROXY=10.0.0.131:3128`

Legen Sie diesen Wert auf den Host-Namen (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys fest, den der Amazon ECS-Agent verwenden soll, um eine Verbindung zum Internet herzustellen. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

`NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Setzen Sie diesen Wert auf `169.254.169.254,169.254.170.2,/var/run/docker.sock`, um die Amazon-EC2-Instance-Metadaten, IAM-Rollen für Aufgaben und Docker-Daemon-Datenverkehr aus dem Proxy zu filtern.

`/etc/systemd/system/ecs.service.d/http-proxy.conf` (nur Amazon Linux 2)

`Environment="HTTP_PROXY=10.0.0.131:3128/"`

Setzen Sie diesen Wert auf den Hostname (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys, sodass mit `ecs-init` eine Internetverbindung aufgebaut werden kann. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

`Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock"`

Setzen Sie diesen Wert auf `169.254.169.254,169.254.170.2,/var/run/docker.sock`, um die Amazon-EC2-Instance-Metadaten, IAM-Rollen für Aufgaben und Docker-Daemon-Datenverkehr aus dem Proxy zu filtern.

`/etc/init/ecs.override` (nur Amazon-Linux-AMI)

`env HTTP_PROXY=10.0.0.131:3128`

Setzen Sie diesen Wert auf den Hostname (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys, sodass mit `ecs-init` eine Internetverbindung aufgebaut werden kann. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

`env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Setzen Sie diesen Wert auf `169.254.169.254,169.254.170.2,/var/run/docker.sock`, um die Amazon-EC2-Instance-Metadaten, IAM-Rollen für Aufgaben und Docker-Daemon-Datenverkehr aus dem Proxy zu filtern.

`/etc/systemd/system/docker.service.d/http-proxy.conf` (nur Amazon Linux 2)

`Environment="HTTP_PROXY=http://10.0.0.131:3128"`

Setzen Sie diesen Wert auf den Hostname (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys, sodass der Docker-Daemon damit eine Internetverbindung aufbauen kann. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

`Environment="NO_PROXY=169.254.169.254"`

Setzen Sie diesen Wert auf `169.254.169.254`, um EC2 Instance-Metadaten aus dem Proxy zu filtern.

`/etc/sysconfig/docker` (Amazon Linux AMI und nur Amazon Linux 2)

`export HTTP_PROXY=http://10.0.0.131:3128`

Setzen Sie diesen Wert auf den Hostname (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys, sodass der Docker-Daemon damit eine Internetverbindung aufbauen kann. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

`export NO_PROXY=169.254.169.254,169.254.170.2`

Setzen Sie diesen Wert auf `169.254.169.254`, um EC2 Instance-Metadaten aus dem Proxy zu filtern.

Die Festlegung dieser Umgebungsvariablen in den oben genannten Dateien wirkt sich nur auf den Amazon ECS Container-Agenten, `ecs-init` und den Docker-Daemon aus. Sie konfigurieren keine anderen Services (z. B. yum) für die Nutzung des Proxy.

Informationen zur Konfiguration des Proxys finden Sie unter Wie richte ich einen HTTP-Proxy für Docker und den Amazon ECS-Container-Agenten in Amazon Linux 2 oder AL2023 ein.

Konfiguration vorinitialisierter Instances für Ihre Amazon ECS Auto Scaling Scaling-Gruppe

Amazon ECS unterstützt Warm-Pools für Amazon EC2 Auto Scaling. Ein Warm-Pool ist eine Gruppe von vorinitialisierten Amazon-EC2-Instances, die bereit sind, in Betrieb genommen zu werden. Wann immer Ihre Anwendung aufskaliert werden muss, verwendet Amazon EC2 Auto Scaling die vorinitialisierten Instances aus dem Warm-Pool, anstatt kalte Instances zu launchen, erlaubt die Ausführung eines endgültigen Initialisierungsprozesses und stellt die Instance dann in Betrieb.

Weitere Informationen zu Warm-Pools und zum Hinzufügen eines Warm-Pools zu Ihrer Auto-Scaling-Gruppe finden Sie unter Warm-Pools für Amazon EC2 Auto Scaling im Benutzerhandbuch zu Amazon EC2 Auto Scaling.

Wenn Sie einen Warm-Pool für eine Auto-Scaling-Gruppe für Amazon ECS erstellen oder aktualisieren, können Sie die Option nicht festlegen, mit der Instances an den Warm-Pool beim Abskalieren zurückgegeben werden (`ReuseOnScaleIn`). Weitere Informationen finden Sie unter put-warm-pool in der AWS Command Line Interface Referenz.

Um Warm-Pools mit Ihrem Amazon-ECS-Cluster zu verwenden, setzen Sie die Agent-Konfigurationsvariable `ECS_WARM_POOLS_CHECK` im Feld User data (Benutzerdaten) der Launchvorlage Ihrer Amazon-EC2-Auto-Scaling-Gruppe auf `true`.

Das folgende Beispiel zeigt, wie die Agent-Konfigurationsvariable im Feld User data (Benutzerdaten) einer Amazon-EC2-Launchvorlage angegeben werden kann. *`MyCluster`*Ersetzen Sie Ihren Cluster durch den Namen unseres Clusters.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_WARM_POOLS_CHECK=true
EOF
```

Diese Variable `ECS_WARM_POOLS_CHECK` wird nur von den Agenten-Versionen `1.59.0` oder höher unterstützt. Weitere Informationen zu der Variablen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Überprüfen des Amazon-ECS-Container-Agenten

Gelegentlich müssen Sie möglicherweise den Amazon ECS-Container-Agenten aktualisieren, um Fehlerkorrekturen und neue Funktionen zu erhalten. Durch die Aktualisierung des Amazon-ECS-Container-Agenten werden laufende Aufgaben oder Services auf der Container-Instance nicht unterbrochen. Der Aktualisierungsvorgang hängt davon ab, ob Ihre Container-Instance mit dem Amazon-ECS-optimierten AMI oder mit einem anderen Betriebssystem gestartet wurde.

> **Note**
>
> Agent-Updates gelten nicht für Windows-Container-Instances. Wir empfehlen, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

Überprüfen der Amazon-ECS-Container-Agent-Version

Sie können die Version des Container-Agenten, der auf Ihren Container-Instances ausgeführt wird prüfen, um festzustellen, ob eine Aktualisierung erforderlich ist. Sie finden die Agenten-Version in der Container-Instance-Ansicht der Amazon-ECS-Konsole. Gehen Sie zum Prüfen Ihrer Agenten-Version wie folgt vor:

Amazon ECS console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre externe Instance registriert ist.
3. Wählen Sie im Navigationsbereich Clusters und danach den Cluster aus, der Ihre externe Instance hostet.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur).
5. Beachten Sie unter Container instances (Container-Instances) die Spalte Agent version (Agent-Version) für Ihre Container-Instances. Wenn die Container-Instance nicht die neueste Container-Agent-Version enthält, erhalten Sie von der Konsole eine Nachricht, die Sie auf die veraltete Version des Agenten hinweist.

   Wenn die Agentenversion veraltet ist, können Sie den Container-Agenten wie folgt aktualisieren:

   - Wenn Ihre Container-Instance das Amazon-ECS-optimierte AMI ausführt, gehen Sie zu Aktualisieren des Amazon-ECS-Container-Agenten auf einem Amazon-ECS-optimierten AMI.
   - Wenn Ihre Container-Instance das Amazon-ECS-optimierte AMI nicht ausführt, gehen Sie zu Manuelles Aktualisieren des Amazon-ECS-Container-Agenten (für Nicht-Amazon-ECS-optimierte AMIs).

   > **Important**
   >
   > Für die Aktualisierung der Amazon-ECS-Agenten-Version mit Versionen vor 1.0.0 auf Ihrem Amazon-ECS-optimierten AMI empfehlen wir Ihnen, zunächst die aktuelle Container-Instance zu beenden und dann eine Instance mit dem aktuellen AMI zu starten. Alle Container-Instances, die eine Vorversion verwenden, sollten durch die aktuelle AMI-Version ersetzt werden. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

Amazon ECS container agent introspection API

Sie können auch den Agenten verwenden, um die Version der Introspektions-API des Amazon-ECS-Container-Agenten in der Container-Instance selbst abzurufen. Weitere Informationen finden Sie unter Introspektion von Amazon ECS-Containern.

So prüfen Sie, ob Ihr Amazon-ECS-Container-Agent die neueste Version in der Introspektions-API verwendet

1. Melden Sie sich bei Ihrer Container-Instance über SSH an.
2. Fragen Sie die Introspektions-API ab.

```
[ec2-user ~]$ curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

> **Note**
>
> Die Introspektions-API hat `Version`-Informationen zur Version 1.0.0 des Amazon-ECS-Container-Agenten hinzugefügt. Wenn `Version` beim Abfragen der Introspektions-API nicht, bzw. die Introspektions-API überhaupt nicht in Ihrem Agenten vorhanden ist, dann bedeutet das, dass die von Ihnen verwendete Version v.0.03 oder älter ist. Sie sollten Ihre Version aktualisieren.

Aktualisieren des Amazon-ECS-Container-Agenten auf einem Amazon-ECS-optimierten AMI

Wenn Sie das Amazon-ECS-optimierte AMI verwenden, stehen Ihnen die folgenden Optionen für die Aktualisierung des Amazon-ECS-Container-Agenten zur Verfügung (Auflistung nach Empfehlungsreihenfolge):

- Beenden Sie Ihre aktuellen Container-Instances und starten Sie die aktuelle Version des Amazon-ECS-optimierten Amazon Linux 2-AMI (entweder manuell oder durch Aktualisierung Ihrer Auto Scaling-Startkonfiguration mit dem neuesten AMI). So erhalten Sie eine neue Container-Instance mit den aktuellen getesteten und validierten Versionen von Amazon Linux, Docker, `ecs-init` und Amazon-ECS-Container-Agent. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.
- Verbinden Sie sich über SSH mit der Instance und aktualisieren Sie das `ecs-init`-Paket (sowie seine Abhängigkeiten) auf die neueste Version. Dieser Vorgang liefert die aktuellsten getesteten und validierten Versionen von Docker und `ecs-init`, die in den Amazon Linux-Repositories

verfügbar sind, sowie die neueste Version des Amazon-ECS-Container-Agenten. Weitere Informationen finden Sie unter So aktualisieren Sie das `ecs-init`-Paket auf dem Amazon-ECS-optimierten AMI.

- Aktualisieren Sie den Container-Agenten mit dem `UpdateContainerAgent` API-Vorgang, entweder über die Konsole oder mit den AWS CLI oder AWS SDKs. Weitere Informationen finden Sie unter Aktualisieren des Amazon-ECS-Container-Agenten mit der `UpdateContainerAgent`-API-Operation.

> **Note**
>
> Agent-Updates gelten nicht für Windows-Container-Instances. Wir empfehlen, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

So aktualisieren Sie das **`ecs-init`**-Paket auf dem Amazon-ECS-optimierten AMI

1. Melden Sie sich bei Ihrer Container-Instance über SSH an.
2. Aktualisieren Sie das `ecs-init`-Paket mit dem folgenden Befehl.

```
sudo yum update -y ecs-init
```

> **Note**
>
> Das `ecs-init`-Paket und der Amazon-ECS-Container-Agent werden sofort aktualisiert. Neuere Versionen von Docker werden jedoch nicht geladen, bis der Docker-Daemon neu gestartet wird. Machen Sie einen Neustart, indem Sie entweder die Instance neu starten oder die folgenden Befehle auf Ihrer Instance ausführen:
>
> - Amazon-ECS-optimiertes Amazon Linux 2-AMI:
>
> ```
> sudo systemctl restart docker
> ```
>
> - Amazon-ECS-optimiertes Amazon Linux AMI:
>
> ```
> sudo service docker restart && sudo start ecs
> ```

Aktualisieren des Amazon-ECS-Container-Agenten mit der **`UpdateContainerAgent`**-API-Operation

> **Important**
>
> Die `UpdateContainerAgent`-API wird nur auf Linux-Varianten des Amazon-ECS-optimierten AMI unterstützt, mit Ausnahme des Amazon-ECS-optimierten Amazon Linux 2 (arm64) AMI. Für Container-Instances, die das Amazon-ECS-optimierte Amazon Linux 2 (arm64) -AAMI verwenden, aktualisieren Sie das `ecs-init`-Paket, um den Agenten zu aktualisieren. Informationen zu Container-Instances, die auf anderen Betriebssystemen laufen, finden Sie unter Manuelles Aktualisieren des Amazon-ECS-Container-Agenten (für Nicht-Amazon-ECS-optimierte AMIs). Sollten Sie Windows-Container-Instances nutzen, empfehlen wir, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

Der `UpdateContainerAgent` API-Prozess beginnt, wenn Sie ein Agent-Update anfordern, entweder über die Konsole oder mit den AWS CLI oder AWS SDKs. Amazon ECS vergleicht Ihre aktuelle Agentenversion mit der neuesten verfügbaren Agentenversion und ob ein Update möglich ist. Wenn keine Aktualisierung verfügbar ist, beispielsweise, wenn der Agent bereits mit der neuesten Version läuft, wird eine `NoUpdateAvailableException` zurückgegeben.

Der oben genannte Aktualisierungsvorgang umfasst folgende Schritte:

`PENDING`

Ein Agent-Aktualisierung ist verfügbar und der Aktualisierungsvorgang wurde gestartet.

`STAGING`

Der Agent hat mit dem Herunterladen der Agent-Aktualisierung begonnen. Wenn der Agent die Aktualisierung nicht herunterladen kann oder wenn der Inhalt der Aktualisierung falsch oder korrupt ist, sendet der Agent eine Benachrichtigung des Fehlers und die Aktualisierung geht in den `FAILED`-Status über.

`STAGED`

Das Herunterladen des Agent ist abgeschlossen und die Agent-Inhalte wurden bestätigt.

`UPDATING`

Der `ecs-init`-Service wird mit der neuen Agenten-Version neu gestartet. Wenn der Agent nicht neu starten kann, geht die Aktualisierung in den `FAILED`-Status über. Andernfalls zeigt der Agent Amazon ECS an, dass die Aktualisierung nicht abgeschlossen wurde.

> **Note**
>
> Agent-Updates gelten nicht für Windows-Container-Instances. Wir empfehlen, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

So aktualisieren Sie im Amazon-ECS-optimierten AMI in der Konsole den Amazon-ECS-Container-Agenten

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre externe Instance registriert ist.
3. Wählen Sie im Navigationsbereich Clusters und dann den Cluster aus.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur).
5. Wählen Sie unter Container-Instances die zu aktualisierenden Instances aus, und wählen Sie dann Aktionen, Agent aktualisieren.

Manuelles Aktualisieren des Amazon-ECS-Container-Agenten (für Nicht-Amazon-ECS-optimierte AMIs)

So aktualisieren Sie manuell den Amazon-ECS-Container-Agenten (für nicht Amazon-ECS-optimierte AMIs)

> **Note**
>
> Agent-Updates gelten nicht für Windows-Container-Instances. Wir empfehlen, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

1. Melden Sie sich bei Ihrer Container-Instance über SSH an.

2. Überprüfen Sie, ob ihr Agent die Umgebungsvariable `ECS_DATADIR` nutzt, um seinen Status zu speichern.

```
ubuntu:~$ docker inspect ecs-agent | grep ECS_DATADIR
```

Ausgabe:

```
"ECS_DATADIR=/data",
```

> **Important**
>
> Wenn der vorherige Befehl die Umgebungsvariable `ECS_DATADIR` nicht zurücksendet, müssen Sie sämtliche Aufgaben, die auf dieser Container-Instance ausgeführt werden, abbrechen, bevor Sie Ihren Agenten aktualisieren. Neuere Agenten mit der Umgebungsvariable `ECS_DATADIR` speichern ihren Status und Sie können sie aktualisieren, während Aufgaben problemlos ausgeführt werden.

3. Halten Sie den Amazon-ECS-Container-Agent an.

```
ubuntu:~$ docker stop ecs-agent
```

4. Löschen Sie den Agent-Container.

```
ubuntu:~$ docker rm ecs-agent
```

5. Stellen Sie sicher, dass das `/etc/ecs`-Verzeichnis und die Konfigurationsdatei für den Amazon-ECS-Container-Agenten unter `/etc/ecs/ecs.config` vorhanden sind.

```
ubuntu:~$ sudo mkdir -p /etc/ecs && sudo touch /etc/ecs/ecs.config
```

6. Bearbeiten Sie die Datei `/etc/ecs/ecs.config` und stellen Sie sicher, dass sie mindestens die folgenden Variablendeklarationen enthält. Wenn Sie nicht möchten, dass Ihre Container-Instance im Default-Cluster registriert wird, legen Sie Ihren Cluster-Namen als einen Wert für `ECS_CLUSTER` fest.

```
ECS_DATADIR=/data
ECS_ENABLE_TASK_IAM_ROLE=true
ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true
ECS_LOGFILE=/log/ecs-agent.log
```

```
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
ECS_LOGLEVEL=info
ECS_CLUSTER=default
```

Weitere Informationen zu diesen und anderen Agenten-Laufzeitoptionen erhalten Sie unterKonfiguration des Amazon-ECS-Container-Agenten.

> **Note**
>
> Sie können Ihre Agenten-Umgebungsvariablen optional in Amazon S3 speichern (diese können in Ihren Container-Instances zum Startzeitpunkt mithilfe von Amazon EC2-Benutzerdaten heruntergeladen werden). Dies empfiehlt sich für sensible Daten, wie beispielsweise Authentifizierungs-Anmeldeinformationen für private Repositorys. Weitere Informationen finden Sie unter Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3 und Verwenden von AWS Nicht-Container-Images in Amazon ECS.

7. Rufen Sie das neueste Image für den Amazon-ECS-Container-Agent von Amazon-Elastic-Container-Registry-Public ab.

```
ubuntu:~$ docker pull public.ecr.aws/ecs/amazon-ecs-agent:latest
```

Ausgabe:

```
Pulling repository amazon/amazon-ecs-agent
a5a56a5e13dc: Download complete
511136ea3c5a: Download complete
9950b5d678a1: Download complete
c48ddcf21b63: Download complete
Status: Image is up to date for amazon/amazon-ecs-agent:latest
```

8. Führen Sie den neuesten Amazon-ECS-Container-Agenten in Ihrer Container-Instance aus.

> **Note**
>
> Verwenden Sie Docker-Neustartrichtlinien oder einen Prozessmanager (wie upstart oder systemd), um den Container-Agenten als Service oder Daemon zu behandeln und sicherzustellen, dass dieser nach dem Beenden wieder gestartet wird. Weitere Informationen finden Sie unter Container automatisch starten und Neustartrichtlinien in

der Docker-Dokumentation. Das Amazon ECS-optimierte AMI verwendet das `ecs-init` RPM für diesen Zweck, und Sie können den Quellcode für dieses RPM unter einsehen. GitHub

Das folgende Beispiel des Agenten-Ausführungsbefehls ist auf mehrere Zeilen verteilt, um die Optionen deutlicher darzustellen. Weitere Informationen zu diesen und anderen Agenten-Laufzeitoptionen erhalten Sie unterKonfiguration des Amazon-ECS-Container-Agenten.

Important

SELinux-fähige Betriebssysteme erfordern die Option `--privileged` in Ihrem Befehl docker run. Für SELinux-fähige Container-Instances empfiehlt es sich, die Option `:Z` zu den `/log` und `/data`-Volumen-Mounts hinzuzufügen. Die Host-Mounts für diese Volumen müssen jedoch vorhanden sein, bevor Sie den Befehl ausführen. Andernfalls erhalten Sie die Fehlermeldung `no such file or directory`. Sollten Sie Schwierigkeiten bei der Ausführung des Amazon-ECS-Agenten auf einer SELinux-fähigen Container-Instance haben, dann verfahren Sie wie folgt:

- Erstellen Sie auf Ihrer Container-Instance Mounting-Punkte für Hoste-Volume.

```
ubuntu:~$ sudo mkdir -p /var/log/ecs /var/lib/ecs/data
```

- Fügen Sie die Option `--privileged` zum unteren Befehl docker run hinzu.
- Fügen Sie die Option `:Z` zu den Container-Volume-Mounts `/log` und `/data` (z. B. `--volume=/var/log/ecs/:/log:Z`) zum unteren Befehl docker run hinzu.

```
ubuntu:~$ sudo docker run --name ecs-agent \
--detach=true \
--restart=on-failure:10 \
--volume=/var/run:/var/run \
--volume=/var/log/ecs/:/log \
--volume=/var/lib/ecs/data:/data \
--volume=/etc/ecs:/etc/ecs \
--volume=/etc/ecs:/etc/ecs/pki \
--net=host \
--env-file=/etc/ecs/ecs.config \
```

```
amazon/amazon-ecs-agent:latest
```

> **Note**
>
> Wenn Sie die Meldung `Error response from daemon: Cannot start container` erhalten, können Sie den fehlerhaften Container mit dem Befehl sudo docker rm ecs-agent löschen und den Agenten neu starten.

Amazon ECS-optimierte Windows-AMIs

Die für Amazon ECS optimierten AMIs sind mit den erforderlichen Komponenten vorkonfiguriert, die Sie zum Ausführen von Amazon-ECS-Workloads benötigen. Sie können zwar Ihr eigenes Container-Instance-AMI erstellen, das die grundlegenden Spezifikationen erfüllt, die für die Ausführung Ihrer containerisierten Workloads auf Amazon ECS erforderlich sind, aber die für Amazon ECS optimierten AMIs sind vorkonfiguriert und wurden von Technikern auf Amazon ECS getestet. AWS Dies ist die einfachste Methode für den Einstieg, mit dem Ihre Container in AWS schnell einsatzbereit werden.

Die Amazon-ECS-optimierten AMI-Metadaten, einschließlich des AMI-Namens, der Version des Amazon-ECS-Container-Agents und der Amazon-ECS-Laufzeitversion, die die Docker-Version enthält, können für jede Variante programmgesteuert abgerufen werden. Weitere Informationen finden Sie unter the section called “Amazon ECS-optimierte Windows AMI-Metadaten abrufen”.

Sie können die Windows AMI Amazon SNS-Themen abonnieren, um benachrichtigt zu werden, wenn ein neues AMI veröffentlicht wird oder eine AMI-Version als privat markiert ist. Weitere Informationen finden Sie unter Abonnieren von Amazon ECS-optimierten Windows AMI-Update-Benachrichtigungen.

> **Important**
>
> Alle ECS-optimierten AMI-Varianten, die nach August produziert wurden, werden von Docker EE (Mirantis) auf Docker CE (Moby-Projekt) migriert.
> Um sicherzustellen, dass alle Kunden standardmäßig stets die neuesten Sicherheitsupdates installiert haben, unterhält Amazon ECS mindestens die letzten drei Windows ECS-optimierten AMIs. Nach der Veröffentlichung neuer für Windows Amazon ECS optimierter AMIs macht Amazon ECS die für Windows Amazon ECS optimierten AMIs, die ältere privat sind. Wenn es ein privates AMI gibt, auf das Sie zugreifen müssen, lassen Sie es uns wissen, indem Sie ein Ticket beim Cloud Support einreichen.

Amazon-ECS-optimierte AMI-Varianten

Die folgenden Windows Server-Varianten des Amazon-ECS-optimierten AMI sind für Ihre Amazon-EC2-Instances verfügbar.

> **Important**
>
> Alle ECS-optimierten AMI-Varianten, die nach August produziert wurden, werden von Docker EE (Mirantis) auf Docker CE (Moby-Projekt) migriert.

- Amazon-ECS-optimierter Windows Server 2022 Full AMI
- Amazon-ECS-optimierter Windows Server 2022 Core AMI
- Amazon-ECS-optimierter Windows Server 2019 Full AMI
- Amazon-ECS-optimierter Windows Server 2019 Core AMI
- Amazon-ECS-optimierter Windows Server 2016 Full AMI

> **Important**
>
> Windows Server 2016 unterstützt nicht die neueste Docker-Version, zum Beispiel 25.x.x. Daher erhalten die Windows Server 2016 Full AMIs keine Sicherheits- oder Bug-Patches für die Docker-Laufzeit. Wir empfehlen Ihnen, auf eine der folgenden Windows-Plattformen zu wechseln:
>
> - Windows Server 2022 Voll
> - Windows Server 2022 Kern
> - Windows Server 2019 Voll
> - Windows Server 2019 Kern

Am 9. August 2022 erreichte das Amazon-ECS-optimierte Windows Server 20H2 Core AMI das Enddatum des Supports. Es werden keine neuen Versionen dieses AMI veröffentlicht. Weitere Informationen finden Sie unter Windows Server-Versionsinformationen.

Windows Server 2022, Windows Server 2019 und Windows Server 2016 sind Long-Term Servicing Channel (LTSC)-Versionen. Windows Server 20H2 ist eine Semi-Annual Channel (SAC)-Version. Weitere Informationen finden Sie unter Windows Server-Versionsinformationen.

Überlegungen

Hier sind einige Dinge, die Sie über Amazon-EC2-Windows-Container und Amazon ECS wissen sollten.

- Windows-Container können nicht auf Linux-Container-Instances ausgeführt werden, und ebenso anders herum. Um sicherzustellen, dass die Aufgaben für Windows und Linux richtig platziert werden, bringen Sie Windows- und Linux-Container-Instances in separaten Clustern unter. Windows-Aufgaben sollten nur in Windows-Clustern platziert werden. Sie können sicherstellen, dass Windows-Aufgabendefinitionen nur auf Windows-Instances platziert werden, indem Sie die folgende Platzierungsbedingung einrichten: `memberOf(ecs.os-type=='windows')`.
- Windows-Container werden für Aufgaben unterstützt, die die Launchtypen EC2 und Fargate verwenden.
- Windows-Container und Windows-Container-Instances unterstützen nicht alle Parameter für Aufgabendefinitionen, die für Linux-Container und Linux-Container-Instances verfügbar sind. Einige Parameter werden gar nicht unterstützt, andere verhalten sich auf Windows und Linux unterschiedlich. Weitere Informationen finden Sie unter Unterschiede bei der Amazon ECS-Aufgabendefinition für EC2-Instances, auf denen Windows ausgeführt wird.
- Sie müssen für das Feature „IAM-Rollen für Aufgaben“ Ihre Windows-Container-Instances so konfigurieren, dass das Feature beim Starten zugelassen wird. Ihre Container müssen den bereitgestellten PowerShell Code ausführen, wenn sie die Funktion verwenden. Weitere Informationen finden Sie unter Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance.
- Das Feature „IAM-Rollen für Aufgaben“ verwendet einen speziellen Proxy, um Anmeldeinformationen für die Container bereitzustellen. Dieser Proxy für Anmeldeinformationen belegt Port 80 auf der Container-Instance. Wenn Sie also „IAM-Rollen für Aufgaben“ verwenden, ist Port 80 für Aufgaben nicht verfügbar. Für Webservice-Container können Sie einen Application Load Balancer und die dynamische Port-Zuweisung verwenden, um Ihren Containern standardmäßige HTTP-Verbindungen über Port 80 bereitzustellen. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.
- Die Docker-Images des Windows-Servers sind groß (9 GiB). Also benötigen Ihre Windows-Container-Instances mehr Speicherplatz als Linux-Container-Instances.
- Um einen Windows-Container auf einem Windows-Server ausführen zu können, muss die Betriebssystemversion des Basis-Image des Containers mit der des Hosts übereinstimmen. Weitere Informationen finden Sie unter Kompatibilität mit Windows-Containern auf der Microsoft-Dokumentations-Website. Wenn auf Ihrem Cluster mehrere Windows-Versionen ausgeführt

werden, können Sie mithilfe der Platzierungsbeschränkung `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)` sicherstellen, dass eine Aufgabe auf einer EC2-Instance platziert wird, die auf derselben Version ausgeführt wird. Weitere Informationen finden Sie unter the section called “Amazon ECS-optimierte Windows AMI-Metadaten abrufen”.

Amazon ECS-optimierte Windows AMI-Metadaten abrufen

Die AMI-ID, der Image-Name, das Betriebssystem, die Container-Agent-Version und die Laufzeitversion für jede Variante der Amazon-ECS-optimierten AMIs können programmgesteuert abgerufen werden, indem Sie die Systems Manager-Parameterspeicher-API abfragen. Weitere Informationen zur Systems Manager Parameter Store-API finden Sie unter GetParametersund GetParametersByPath.

> **Note**
>
> Ihr administrativer Benutzer muss über die folgenden IAM-Berechtigungen verfügen, um die Amazon-ECS-optimierten AMI-Metadaten abzurufen. Diese Berechtigungen wurden der `AmazonECS_FullAccess`-IAM-Richtlinie hinzugefügt.
>
> - ssm: GetParameters
> - ssm: GetParameter
> - ssm: GetParameters ByPath

Systems Manager Parameterspeicher-Parameterformat

> **Note**
>
> Die folgenden API-Parameter von Systems Manager Parameter Store sind veraltet und sollten nicht zum Abrufen der neuesten Windows-AMIs verwendet werden:
>
> - `/aws/service/ecs/optimized-ami/windows_server/2016/english/full/recommended/image_id`
> - `/aws/service/ecs/optimized-ami/windows_server/2019/english/full/recommended/image_id`

Im Folgenden ist das Format des Parameternamens für jede Amazon-ECS-optimierte AMI-Variante aufgeführt.

- Metadaten von Windows Server 2022 Full AMI:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

- Metadaten von Windows Server 2022 Core AMI:

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

- Windows Server 2019 Full AMI-Metadaten:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

- Windows Server 2019 Core AMI-Metadaten:

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

- Windows Server 2016 Full AMI-Metadaten:

```
/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Das folgende Parameter-Namensformat ruft die Metadaten des neuesten stabilen Windows Server 2019 Full AMI ab

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized
```

Nachstehend finden Sie ein Beispiel des JSON-Objekts, das für den Parameterwert zurückgegeben wird.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized",
            "Type": "String",
```

```
            "Value": "{\"image_name\":\"Windows_Server-2019-English-Full-
ECS_Optimized-2023.06.13\",\"image_id\":\"ami-0debc1fb48e4aee16\",\"ecs_runtime_version
\":\"Docker (CE) version 20.10.21\",\"ecs_agent_version\":\"1.72.0\"}",
            "Version": 58,
            "LastModifiedDate": "2023-06-22T19:37:37.841000-04:00",
            "ARN": "arn:aws:ssm:us-east-1::parameter/aws/service/ami-windows-latest/
Windows_Server-2019-English-Full-ECS_Optimized",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```


Jedes der Felder oben in der Ausgabe steht zur Abfrage als Sub-Parameter zur Verfügung. Erstellen Sie den Parameterpfad für einen Sub-Parameter, indem Sie den Sub-Parameternamen an den Pfad für das ausgewählte AMI anhängen. Die folgenden Sub-Parameter sind verfügbar:

- `schema_version`
- `image_id`
- `image_name`
- `os`
- `ecs_agent_version`
- `ecs_runtime_version`

Beispiele

Die folgenden Beispiele zeigen, wie Sie die Metadaten für jede Amazon-ECS-optimierte AMI-Variante abrufen können.

Abrufen der Metadaten des neuesten stabilen Amazon-ECS-optimierten AMI

Sie können das neueste stabile Amazon ECS-optimierte AMI AWS CLI mit den folgenden AWS CLI Befehlen abrufen.

- Für das Amazon-ECS-optimierte Windows Server 2022 Full AMI:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-
English-Full-ECS_Optimized --region us-east-1
```

- Für das Amazon-ECS-optimierte Windows Server 2022 Core AMI:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized --region us-east-1
```

- Für das Amazon-ECS-optimierte Windows Server 2019 Full AMI:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized --region us-east-1
```

- Für das Amazon-ECS-optimierte Windows Server 2019 Core AMI:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized --region us-east-1
```

- Für das Amazon-ECS-optimierte Windows Server 2016 Full AMI:

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized --region us-east-1
```

Verwenden des neuesten empfohlenen Amazon ECS-optimierten AMI in einer Vorlage AWS CloudFormation

Sie können auf das neueste empfohlene Amazon-ECS-optimierte AMI in einer AWS CloudFormation -Vorlage verweisen, indem Sie auf den Systems Manager Parameterspeichernamen verweisen.

```
Parameters:
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized/image_id
```

Abonnieren von Amazon ECS-optimierten Windows AMI-Update-Benachrichtigungen

AWS bietet zwei Amazon SNS SNS-Themen-ARNs für Benachrichtigungen im Zusammenhang mit den Windows Server-AMIs. Ein Thema sendet Update-Benachrichtigungen, wenn neue Windows Server-AMIs veröffentlicht werden. Das andere Thema sendet Benachrichtigungen, wenn zuvor freigegebene Windows Server-AMIs privat gemacht werden. Diese Themen beziehen sich zwar nicht auf die von Amazon ECS optimierten Windows-AMIs, da die für Amazon ECS optimierten Windows-AMIs demselben Release-Zeitplan folgen, können Sie diese Benachrichtigungen jedoch als Hinweis

darauf verwenden, wann neue Amazon-ECS-optimierte Windows-AMIs aktualisiert werden. Weitere Informationen zum Abonnieren von Windows AMI-Benachrichtigungen finden Sie unter Abonnieren von Windows AMI-Benachrichtigungen im Amazon EC2 EC2-Benutzerhandbuch.

> **Note**
>
> Ihr Benutzer oder die Ihrem Benutzer zugeordnete Rolle muss über die `sns::subscribe`-IAM-Berechtigung verfügen, um ein Amazon SNS-Thema zu abonnieren.

Amazon ECS-optimierte Windows AMI-Versionen

Sehen Sie sich die aktuellen und früheren Versionen der Amazon ECS-optimierten AMIs und die entsprechenden Versionen des Amazon ECS-Container-Agenten, Docker und des Pakets an. `ecs-init`

Die Amazon-ECS-optimierten AMI-Metadaten, einschließlich der AMI-ID für jede Variante, können programmgesteuert abgerufen werden. Weitere Informationen finden Sie unter the section called “Amazon ECS-optimierte Windows AMI-Metadaten abrufen”.

Auf den folgenden Registerkarten wird eine Liste der für Windows Amazon ECS optimierten AMIs Versionen angezeigt. Einzelheiten zum Verweisen auf den Systems Manager Manager-Parameter Store-Parameter in einer AWS CloudFormation Vorlage finden Sie unterVerwenden des neuesten empfohlenen Amazon ECS-optimierten AMI in einer Vorlage AWS CloudFormation.

> **Important**
>
> Um sicherzustellen, dass alle Kunden standardmäßig stets die neuesten Sicherheitsupdates installiert haben, unterhält Amazon ECS mindestens die letzten drei Windows ECS-optimierten AMIs. Nach der Veröffentlichung neuer für Windows Amazon ECS optimierter AMIs macht Amazon ECS die für Windows Amazon ECS optimierten AMIs, die ältere privat sind. Wenn es ein privates AMI gibt, auf das Sie zugreifen müssen, lassen Sie es uns wissen, indem Sie ein Ticket beim Cloud Support einreichen.
> Windows Server 2016 unterstützt nicht die neueste Docker-Version, z. B. 25.x.x. Daher erhalten die Windows Server 2016 Full AMIs keine Sicherheits- oder Bug-Patches für die Docker-Laufzeit. Wir empfehlen Ihnen, auf eine der folgenden Windows-Plattformen zu wechseln:
>
> - Windows Server 2022 Voll

- Windows Server 2022 Kern
- Windows Server 2019 Voll
- Windows Server 2019 Kern

Windows Server 2022 Full AMI versions

Die folgende Tabelle listet die aktuellen und vorherigen Versionen des Amazon-ECS-optimierten Windows Server 2022 Full AMI und ihre entsprechenden Versionen des Amazon-ECS-Container-Agenten und -Dockers auf.

Amazon-ECS-optimierter Windows Server 2022 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-Englisch-Voll-ECS_Optimized-2024.05.14	`1.82.3`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Voll-ECS_Optimiert-2024.04.09	`1.82.2`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Voll-ECS_Optimiert-2024.03.12	`1.82.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Voll-ECS_Optimiert-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Voll-ECS	`1.79.2`	`20.10.23 (Docker CE)`	Öffentlich

Amazon-ECS-optimierter Windows Server 2022 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
_Optimiert-2024.01.09			
Windows_Server-2022-Englisch-Voll-ECS_Optimiert-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2022-Englisch-Voll-ECS_Optimiert-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.08.09	`1.74.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.07.11	`1.73.1`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2022 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Full-ECS_Optimized-2023.06.13	`1.72.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.05.18	`1.71.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.04.18	`1.70.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.02.21	`1.68.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2022 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Full-ECS_Optimized-2022.11.09	1.65.1	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.10.12	1.64.0	20.10.17 (Docker CE)	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.09.22	1.63.1	20.10.17 (Docker CE)	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.09.14	1.62.2	20.10.17 (Docker CE)	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privat

Amazon-ECS-optimierter Windows Server 2022 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Full-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privat
Windows_Server-2022-English-Full-ECS_Optimized-2021.009.23	1.55.3	20.10.7	Privat

Verwenden Sie den folgenden AWS CLI Befehl, um das aktuelle Amazon ECS-optimierte Windows Server 2022 Full AMI abzurufen.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

Windows Server 2022 Core AMI versions

Die folgende Tabelle listet die aktuellen und vorherigen Versionen des Amazon-ECS-optimierten Windows Server 2022 Core AMI und ihre entsprechenden Versionen des Container-Agenten und Dockers von Amazon ECS auf.

Amazon-ECS-optimierter Windows Server 2022 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Core-ECS_Optimized-2024.05.14	`1.82.3`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Core-ECS_Optimiert-2024.04.09	`1.82.2`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Core-ECS_Optimiert-2024.03.12	`1.82.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Core-ECS_Optimiert-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-202-Englisch-Core-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2022-Englisch-Core-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2022-Englisch-Core-EC	`1.79.0`	`20.10.23 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2022 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
S_Optimized-2023.11.14			
Windows_Server-2022-English-Core-ECS_Optimized-2023.10.11	1.77.0	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.09.15	1.75.3	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.08.09	1.74.1	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.07.11	1.73.1	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privat

Amazon-ECS-optimierter Windows Server 2022 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Core-ECS_Optimized-2023.04.18	`1.70.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.02.21	`1.68.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2022 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Core-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.09.14	`1.62.2`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.08.15	`1.62.1`	`20.10.9`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.07.13	`1.61.3`	`20.10.9`	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2022.06.15	`1.61.2`	`20.10.9`	Privat

Amazon-ECS-optimierter Windows Server 2022 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2022-English-Core-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privat
Windows_Server-2022-English-Core-ECS_Optimized-2021.009.23	1.55.3	20.10.7	Privat

Verwenden Sie den folgenden AWS CLI Befehl, um das aktuelle Amazon ECS-optimierte Windows Server 2022 Full AMI abzurufen.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

Windows Server 2019 Full AMI versions

Die folgende Tabelle listet die aktuellen und vorherigen Versionen des Amazon-ECS-optimierten Windows Server 2019 Full AMI und ihre entsprechenden Versionen von Amazon-ECS-Container-Agent und Docker auf.

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-Englisch-Voll-ECS_Optimized-2024.05.14	`1.82.3`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Voll-ECS_Optimiert-2024.04.09	`1.82.2`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Voll-ECS_Optimiert-2024.03.12	`1.82.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Voll-ECS_Optimiert-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Voll-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Voll-ECS_Optimiert-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2019-Englisch-Voll-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.08.09	`1.74.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.07.11	`1.73.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.06.13	`1.72.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.05.18	`1.71.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.04.18	`1.70.2`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.02.21	`1.68.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2022.09.14	1.62.2	20.10.17 (Docker CE)	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2021.009.23	1.55.3	20.10.7	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.08.12	1.55.0	20.10.6	Öffentlich
Windows_Server-2019-English-Full-ECS_Optimized-2021.07.13	1.54.02	20.10.6	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.07.08	1.54.0	20.10.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.06.11	1.53.0	20.10.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.05.21	1.52.2	20.10.4	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.04.14	1.51.0	20.10.0	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2021.03.11	1.50.2	19.03.14	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.02.10	1.50.0	19.03.14	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2021.01.13	1.49.0	19.03.14	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.11.18	1.48.0	19.03.13	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.11.06	1.47.0	19.03.11	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.10.14	1.45.0	19.03.11	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.08.12	1.43.0	19.03.11	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2020.07.15	1.41.1	19.03.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.06.11	1.40.0	19.03.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.05.14	1.39.0	19.03.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2020.01.15	1.35.0	19.03.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.12.16	1.34.0	19.03.5	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.11.25	1.34.0	19.03.4	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.11.13	1.32.1	19.03.4	Privat

Amazon-ECS-optimierter Windows Server 2019 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Full-ECS_Optimized-2019.10.09	1.32.0	19.03.2	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.09.11	1.30.0	19.03.1	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.08.16	1.29.1	19.03.1	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.07.19	1.29.0	18.09.8	Privat
Windows_Server-2019-English-Full-ECS_Optimized-2019.05.10	1.27.0	18.09.4	Privat

Verwenden Sie den folgenden AWS CLI Befehl, um das aktuelle Amazon ECS-optimierte Windows Server 2019 Full AMI abzurufen.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

Windows Server 2019 Core AMI versions

Important

Die folgende Tabelle listet die aktuellen und vorherigen Versionen des Amazon-ECS-optimierten Windows Server 2019 Core AMI und ihre entsprechenden Versionen von Amazon-ECS-Container-Agent und Docker auf.

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-Englisch-Core-ECS_Optimized-2024.05.14	`1.82.3`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Core-ECS_Optimiert-2024.04.09	`1.82.2`	`25.0.3 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Core-ECS_Optimiert-2024.03.12	`1.82.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Core-ECS_Optimiert-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Öffentlich

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-Englisch-Core-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2019-Englisch-Core-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2019-Englisch-Core-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2023.08.09	`1.74.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.07.11	`1.73.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.06.13	`1.72.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.05.18	`1.71.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.04.18	`1.70.2`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.02.21	`1.68.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.09.14	`1.62.2`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.08.15	`1.62.1`	`20.10.9`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.07.13	`1.61.3`	`20.10.9`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2022.06.15	`1.61.2`	`20.10.9`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2022.01.18	`1.57.1`	`20.10.9`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.12.16	`1.57.1`	`20.10.7`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.11.11	`1.57.0`	`20.10.7`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.09.23	`1.55.3`	`20.10.7`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2021.08.12	`1.55.0`	`20.10.6`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.07.13	`1.54.02`	`20.10.6`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.07.08	`1.54.0`	`20.10.6`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.06.11	`1.53.0`	`20.10.5`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.05.21	`1.52.2`	`20.10.4`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2021.04.14	`1.51.0`	`20.10.0`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.03.11	`1.50.2`	`19.03.14`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.02.10	`1.50.0`	`19.03.14`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2021.01.13	`1.49.0`	`19.03.14`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.11.18	`1.48.0`	`19.03.13`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2020.11.06	`1.47.0`	`19.03.11`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.10.14	`1.45.0`	`19.03.11`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.09.09	`1.44.3`	`19.03.11`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.08.12	`1.43.0`	`19.03.11`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.07.15	`1.41.1`	`19.03.5`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2020.06.11	`1.40.0`	`19.03.5`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.05.14	`1.39.0`	`19.03.5`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2020.01.15	`1.35.0`	`19.03.5`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2019.12.16	`1.34.0`	`19.03.5`	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2019.11.25	`1.34.0`	`19.03.4`	Privat

Amazon-ECS-optimierter Windows Server 2019 Core AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2019-English-Core-ECS_Optimized-2019.11.13	1.32.1	19.03.4	Privat
Windows_Server-2019-English-Core-ECS_Optimized-2019.10.09	1.32.0	19.03.2	Privat

Verwenden Sie den folgenden AWS CLI Befehl, um das aktuelle Amazon ECS-optimierte Windows Server 2019 Full AMI abzurufen.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

Windows Server 2016 Full AMI versions

Important

Windows Server 2016 unterstützt nicht die neueste Docker-Version, zum Beispiel 25.x.x. Daher erhalten die Windows Server 2016 Full AMIs keine Sicherheits- oder Bug-Patches für die Docker-Laufzeit. Wir empfehlen Ihnen, auf eine der folgenden Windows-Plattformen zu wechseln:

- Windows Server 2022 Voll
- Windows Server 2022 Kern
- Windows Server 2019 Voll

- Windows Server 2019 Kern

Die folgende Tabelle listet die aktuellen und vorherigen Versionen des Amazon-ECS-optimierten Windows Server 2016 Full AMI und ihre entsprechenden Versionen von Amazon-ECS-Container-Agent und Docker auf.

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-Englisch-Voll-ECS_Optimized-2024.03.12	`1.82.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2016-Englisch-Voll-ECS_Optimiert-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2016-Englisch-Voll-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2016-Englisch-Voll-ECS_Optimiert-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2016-Englisch-Voll-ECS_Optimiert-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Öffentlich
Windows_Server-2016-English-Full-ECS	`1.77.0`	`20.10.21 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
_Optimized-2023.10.11			
Windows_Server-2016-English-Full-ECS_Optimized-2023.09.15	1.75.3	20.10.21 (Docker CE)	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.08.09	1.74.1	20.10.21 (Docker CE)	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.07.11	1.73.1	20.10.21 (Docker CE)	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.04.18	1.70.2	20.10.21 (Docker CE)	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.02.21	`1.68.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2022.09.14	`1.62.2`	`20.10.17 (Docker CE)`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.08.15	`1.62.1`	`20.10.9`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.07.13	`1.61.3`	`20.10.9`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.06.15	`1.61.2`	`20.10.9`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2022.01.18	`1.57.1`	`20.10.9`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.12.16	`1.57.1`	`20.10.7`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.11.11	`1.57.0`	`20.10.7`	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2021.09.23	1.55.3	20.10.7	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.08.12	1.55.0	20.10.6	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.07.13	1.54.02	20.10.6	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.07.08	1.54.0	20.10.5	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.06.11	1.53.0	20.10.5	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.05.21	1.52.2	20.10.4	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.04.14	1.51.0	20.10.0	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2021.03.11	1.50.2	19.03.14	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.02.10	1.50.0	19.03.14	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2021.01.13	1.49.0	19.03.14	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.11.18	1.48.0	19.03.13	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.11.06	1.47.0	19.03.11	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.10.14	1.45.0	19.03.12	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.09.09	1.44.3	19.03.11	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2020.08.12	`1.43.0`	`19.03.11`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.07.15	`1.41.1`	`19.03.5`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.06.11	`1.40.0`	`19.03.5`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.05.14	`1.39.0`	`19.03.5`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2020.01.15	`1.35.0`	`19.03.5`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.12.16	`1.34.0`	`19.03.5`	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.11.25	`1.34.0`	`19.03.4`	Privat

Amazon-ECS-optimierter Windows Server 2016 Full AMI	Version des Amazon-ECS-Container-Agenten	Docker-Version	Sichtbarkeit
Windows_Server-2016-English-Full-ECS_Optimized-2019.11.13	1.32.1	19.03.4	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.10.09	1.32.0	19.03.2	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.09.11	1.30.0	19.03.1	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.08.16	1.29.1	19.03.1	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.07.19	1.29.0	18.09.8	Privat
Windows_Server-2016-English-Full-ECS_Optimized-2019.03.07	1.26.0	18.03.1	Privat

Verwenden Sie das folgende AWS CLI Amazon ECS-optimierte Windows Server 2016 Full AMI.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Erstellen eines eigenen Amazon-ECS-optimierten Windows-AMI

Verwenden Sie EC2 Image Builder, um Ihr eigenes benutzerdefiniertes Amazon ECS-optimiertes Windows-AMI zu erstellen. Das macht es einfach, ein Windows AMI mit Ihrer eigenen Lizenz auf Amazon ECS zu verwenden. Amazon ECS bietet eine verwaltete Image Builder-Komponente, die die Systemkonfiguration bereitstellt, die zum Ausführen von Windows-Instances zum Hosten Ihrer Container erforderlich ist. Jede von Amazon ECS verwaltete Komponente enthält einen bestimmten Container-Agent und eine Docker-Version. Sie können Ihr Image so anpassen, dass es entweder die neueste von Amazon ECS verwaltete Komponente verwendet oder wenn ein älterer Container-Agent oder eine Docker-Version benötigt wird, können Sie eine andere Komponente angeben.

Einen vollständigen Walkthrough für die Verwendung von EC2 Image Builder finden Sie unter Erste Schritte mit EC2 Image Builder im Benutzerhandbuch für EC2 Image Builder.

Wenn Sie Ihr eigenes Amazon-ECS-optimiertes Windows AMI mit EC2 Image Builder erstellen, erstellen Sie ein Image-Rezept. Ihr Image-Rezept muss die folgenden Anforderungen erfüllen:

- Das Quell-Image sollte auf Windows Server 2019 Core, Windows Server 2019 Full, Windows Server 2022 Core oder Windows Server 2022 Full basieren. Jedes andere Windows-Betriebssystem wird nicht unterstützt und ist möglicherweise nicht mit der Komponente kompatibel.
- Bei der Angabe der Erstellungs-Komponenten ist die `ecs-optimized-ami-windows`-Komponente erforderlich. Die `update-windows`-Komponente wird empfohlen, wodurch sichergestellt wird, dass das Image die neuesten Sicherheitsupdates enthält.

  Um eine andere Komponentenversion anzugeben, erweitern Sie das Kontrollkästchen Optionen für die Versionierung und geben Sie die Komponentenversion an, die Sie verwenden möchten. Weitere Informationen finden Sie unter Auflisten der `ecs-optimized-ami-windows`-Komponentenversionen.

Auflisten der **`ecs-optimized-ami-windows`**-Komponentenversionen

Wenn Sie ein EC2 Image Builder Rezept erstellen und die `ecs-optimized-ami-windows`-Komponente angeben, können Sie entweder die Standardoption verwenden oder eine bestimmte Komponentenversion angeben. Um zu bestimmen, welche Komponentenversionen verfügbar sind,

sowie welcher Amazon-ECS-Container-Agent und Docker-Versionen in der Komponente enthalten sind, benutzen Sie AWS Management Console.

Um die verfügbaren **`ecs-optimized-ami-windows`**-Komponentenversionen aufzulisten

1. Öffnen Sie die EC2 Image Builder-Konsole unter https://console.aws.amazon.com/imagebuilder/.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Sie Ihr Image erstellen.
3. Wählen Sie im Navigationsbereich unter dem Menü Gespeicherte Konfigurationen Komponenten aus.
4. Auf der Komponenten-Seite geben Sie ins Suchfeld `ecs-optimized-ami-windows` ein, ziehen das Qualifikationsmenü herunter und wählen Schnellstart (von Amazon verwaltet) aus.
5. Verwenden Sie die Spalte Beschreibung, um die Komponentenversion mit dem Amazon-ECS-Container-Agent und der Docker-Version zu ermitteln, die Ihr Image benötigt.

Verwaltung von Amazon ECS Windows-Container-Instances

Wenn Sie EC2-Instances für Ihre Amazon ECS-Workloads verwenden, sind Sie für die Wartung der Instances verantwortlich.

Agent-Updates gelten nicht für Windows-Container-Instances. Wir empfehlen, dass Sie die neuen Container-Instances starten, um die Agent-Version in Ihren Windows-Clustern zu aktualisieren.

Verwaltungsverfahren

- Starten einer Amazon ECS Windows-Container-Instance
- Bootstrapping von Amazon ECS-Windows-Container-Instances zur Datenübergabe
- Verwenden eines HTTP-Proxys für Amazon ECS Windows-Container-Instances
- Konfiguration von Amazon ECS-Windows-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen

Starten einer Amazon ECS Windows-Container-Instance

Ihre Amazon-ECS-Container-Instances werden mit der Amazon EC2 Konsole erstellt. Bevor Sie beginnen, sollten Sie sicherstellen, dass Sie die in Einrichtung für die Verwendung von Amazon ECS beschriebenen Schritte ausgeführt haben.

Weitere Informationen zum Startassistenten finden Sie unter Starten einer Instance mithilfe des Assistenten zum Starten einer Instance im Amazon EC2 EC2-Benutzerhandbuch.

Sie können den neuen Amazon-EC2-Assistenten verwenden, um eine Instance zu starten. Sie können die folgende Liste für die Parameter verwenden und die Parameter standardmäßig nicht aufgeführt lassen. Die folgenden Anweisungen führen Sie durch jede Parametergruppe.

Verfahren

Bevor Sie beginnen, führen Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS aus.

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. In der Navigationsleiste oben auf dem Bildschirm wird die aktuelle AWS Region angezeigt (z. B. USA Ost (Ohio)). Wählen Sie eine Region aus, in der die Instance gestartet werden soll. Die Auswahl ist wichtig, da nur bestimmte Amazon EC2-Ressourcen zwischen Regionen geteilt werden können.
3. Wählen Sie im Dashboard der Amazon EC2-Konsole die Option Instance starten aus.

Name und Tags

Der Instance-Name ist ein Tag, wobei der Schlüssel Name ist und es sich bei dem Wert um den von Ihnen angegebenen Namen handelt. Sie können Instance, Volumes und elastische Grafiken markieren. Bei Spot-Instances können Sie nur die Spot-Instance-Anforderung mit Tags (Markierungen) versehen.

Die Angabe eines Instance-Namens und zusätzlicher Tags ist optional.

- Geben Sie unter Name einen beschreibenden Namen für die Instance ein. Wenn Sie keinen Namen angeben, kann die Instance anhand der ID identifiziert werden, die beim Starten der Instance automatisch generiert wird.
- Wenn Sie zusätzliche Tags hinzufügen möchten, wählen Sie Add additional tags (Zusätzliche Tags hinzufügen) aus. Klicken Sie auf Tag hinzufügen, geben Sie dann einen Schlüssel und einen Wert ein und wählen Sie den Ressourcentyp aus, den Sie markieren möchten. Wählen Sie für jedes weitere Tag Add another Tag (Weiteres Tag hinzufügen) aus.

Anwendungs- und Betriebssystem-Images (Amazon Machine Image)

Ein Amazon Machine Image (AMI) enthält die Informationen, die zum Starten einer Instance erforderlich sind. Ein AMI könnte beispielsweise die für die Funktion als Webserver erforderliche Software (z. B. Apache) und Ihre Website enthalten.

Informationen zu den neuesten Amazon-ECS-optimierten AMIs und deren Werten finden Sie unter Windows-Amazon-ECS-optimiertes AMI.

Verwenden Sie die Suchleiste, um ein geeignetes Amazon ECS-optimiertes AMI zu finden, das von veröffentlicht wurde. AWS

1. Geben Sie basierend auf Ihren Anforderungen eines der folgenden AMIs in die Search (Such)-Leiste ein und drücken Sie die Enter (Eingabe)-Taste.
   - Windows_Server-2022-English-Full-ECS_Optimized
   - Windows_Server-2022-English-Core-ECS_Optimized
   - Windows_Server-2019-English-Full-ECS_Optimized
   - Windows_Server-2019-English-Core-ECS_Optimized
   - Windows_Server-2016-English-Full-ECS_Optimized
2. Wählen Sie auf der Seite Amazon Machine Image (AMI) wählen die Kategorie Community-AMIs aus.
3. Wählen Sie in der angezeigten Liste ein von Microsoft verifiziertes AMI mit dem letzten Veröffentlichungsdatum aus, und klicken Sie auf Auswählen.

Instance-Typ

Der Instance-Typ definiert die Hardware-Konfiguration und Größe der Instance. Größere Instance-Typen haben mehr CPU und Arbeitsspeicher. Weitere Informationen finden Sie unter Instance-Typen.

- Wählen Sie unter Instance-Typ den Instance-Typ für die Instance aus.

  Von dem von Ihnen ausgewählten Instance-Typ ist abhängig, welche Ressourcen zur Ausführung Ihrer Aufgaben verfügbar sind.

Schlüsselpaar (Anmeldung)

Wählen Sie für Schlüsselpaarname ein vorhandenes Schlüsselpaar aus oder wählen Sie Neues Schlüsselpaar erstellen, um ein neues zu erstellen.

> **Important**
>
> Wenn Sie die Option Proceed without key pair (Not recommended) (Ohne Schlüsselpaar fortfahren (Nicht empfohlen)) auswählen, können Sie keine Verbindung zur Instance

herstellen, es sei denn, Sie wählen ein AMI aus, das entsprechend konfiguriert ist, um Benutzern eine andere Anmeldemöglichkeit zu erlauben.

Network settings (Netzwerkeinstellungen)

Konfigurieren Sie die Netzwerkeinstellungen nach Bedarf.

- Networking platform (Netzwerkplattform): Wählen Sie Virtual Private Cloud (VPC) aus und geben Sie dann im Abschnitt Network interfaces (Netzwerkschnittstellen) das Subnetz an.
- VPC: Wählen Sie eine vorhandene VPC aus, in der die Sicherheitsgruppe erstellt werden soll.
- Subnetz: Sie können eine Instance in einem Subnetz starten, das einer Availability Zone, einer Local Zone, Wavelength-Zone oder einem Outpost zugeordnet ist.

  Um die Instance in einer Availability Zone zu starten, wählen Sie das Subnetz aus, in dem die Instance gestartet werden soll. Um ein neues Subnetz zu erstellen, wählen Sie Create new subnet aus, um die Amazon VPC-Konsole aufzurufen. Wenn Sie fertig sind, kehren Sie zum Launch Instance Wizard zurück und wählen Sie das Symbol zum Aktualisieren, um Ihr Subnetz in die Liste zu laden.

  Um die Instance in einer Local Zone zu starten, wählen Sie ein Subnetz aus, das Sie in der Local Zone erstellt haben.

  Um eine Instance in einem Outpost zu starten, wählen Sie ein Subnetz in einer VPC aus, die Sie dem Outpost zugeordnet haben.
- Auto-assign Public IP (Öffentliche IP automatisch zuweisen): Wenn Ihre Instance über das öffentliche Internet zugänglich sein soll, muss das Feld Auto-assign Public IP (Öffentliche IP automatisch zuweisen) auf Enable (Aktivieren) festgelegt sein. Falls nicht, setzen Sie dieses Feld auf Disable (Deaktivieren).

> **Note**
>
> Container-Instances benötigen Zugriff, um mit dem Amazon-ECS-Service-Endpunkt zu kommunizieren. Dies kann über einen Schnittstellen-VPC-Endpunkt oder über Ihre Container-Instances mit öffentlichen IP-Adressen erfolgen.
> Weitere Informationen zu Schnittstellen-VPC-Endpunkten finden Sie unter Amazon ECS und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).

Wenn Sie keinen Schnittstellen-VPC-Endpunkt konfiguriert haben und Ihre Container-Instances keine öffentlichen IP-Adressen haben, müssen Sie Netzwerkadressübersetzung (Network Address Translation, NAT) verwenden, um diesen Zugriff zur Verfügung zu stellen. Weitere Informationen finden Sie unter NAT-Gateways im Amazon-VPC-Benutzerhandbuch und unter Verwenden eines HTTP-Proxys für Amazon ECS Linux-Container-Instances in dieser Anleitung.

- Firewall (security groups) (Firewall (Sicherheitsgruppen)): Verwenden Sie eine Sicherheitsgruppe, um Firewallregeln für Ihre Container-Instance zu definieren. Diese Regeln legen fest, welcher eingehende Netzwerkverkehr an Ihre Container-Instance weitergeleitet wird. Der gesamte übrige Datenverkehr wird ignoriert.
  - Um eine vorhandene Sicherheitsgruppe auszuwählen, wählen Sie Select existing security group (Vorhandene Sicherheitsgruppe auswählen) und wählen Sie die Sicherheitsgruppe aus, die Sie in Einrichtung für die Verwendung von Amazon ECS erstellt haben.

Speicher konfigurieren

Die von Ihnen ausgewählte AMI beinhaltet ein oder mehrere Speicher-Volumes, einschließlich eines Root-Volumes. Sie können zusätzliche Volumes angeben, die an die Instance angehängt werden sollen.

Sie können die Ansicht Simple (Einfach) verwenden.

- Storage type (Speichertyp): Konfigurieren Sie den Speicher für Ihre Container-Instance.

  Wenn Sie das Amazon-ECS-optimierte Amazon Linux 2-AMI verwenden, ist für Ihre Instance ein einzelnes 30 GiB-Volume konfiguriert, das zwischen dem Betriebssystem und Docker geteilt wird.

  Wenn Sie das Amazon-ECS-optimierte AMI verwenden, erhält Ihre Instance zwei Volumes konfiguriert. Das Root-Volume ist für die Nutzung durch das Betriebssystem und das zweite Amazon EBS-Volume (zugeordnet zu `/dev/xvdcz`) ist für die Nutzung durch Docker vorgesehen.

  Sie haben auch die Möglichkeit, die Volume-Größen für Ihre Instance entsprechend den Anforderungen Ihrer Anwendung zu erhöhen oder zu verringern.

Erweiterte Details

Erweitern Sie für Advanced details (Erweiterte Details) den Bereich zur Ansicht der Felder und geben Sie zusätzliche Parameter für die Instance an.

- Purchasing option (Kaufoption): Wählen Sie Request Spot instances (Spot-Instances anfordern) aus, um Spot-Instances anzufordern. Außerdem müssen Sie in den anderen Felder im Zusammenhang mit Spot-Instances Werte festlegen. Weitere Informationen finden Sie unter Spot-Instance-Anforderungen.

  > **Note**
  >
  > Wenn Sie Spot-Instances verwenden und die Meldung `Not available` angezeigt wird, muss möglicherweise ein anderer Instance-Typ angegeben werden.

  .

- IAM instance profile (IAM-Instance-Profil): Wählen Sie die IAM-Rolle Ihrer Container-Instance aus. Diese heißt normalerweise `ecsInstanceRole`.

  > **Important**
  >
  > Wenn Sie Ihre Container-Instance nicht mit den richtigen IAM-Berechtigungen starten, kann Ihr Amazon-ECS-Agent keine Verbindung mir Ihrem Cluster herstellen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

- (Optional) User data (Benutzerdaten): Konfigurieren Sie Ihre Amazon-ECS-Container-Instance mit Benutzerdaten, wie z. B. den Agenten-Umgebungsvariablen aus Konfiguration des Amazon-ECS-Container-Agenten. Amazon EC2-Benutzerdaten-Skripts werden nur einmal ausgeführt, wenn die Instance erstmals gestartet wird. Im Folgenden finden Sie einige gängige Beispiele dafür, wofür Benutzerdaten verwendet werden:

  - Standardmäßig startet Ihre Container-Instance in Ihrem Standard-Cluster. Für einen Start in einem nicht standardmäßigen Cluster wählen Sie die Liste Advanced Details. Fügen Sie dann das folgende Skript in das Feld User data ein und ersetzen Sie *`your_cluster_name`* durch den Namen Ihres Clusters.

    `EnableTaskIAMRole` aktiviert das Task-IAM-Rollenfeature für die Aufgaben.

Darüber hinaus sind die folgenden Optionen verfügbar, wenn Sie den `awsvpc`-Netzwerkmodus verwenden.

- `EnableTaskENI`: Dieses Flag aktiviert das Aufgabennetzwerk und ist erforderlich, wenn Sie den `awsvpc`-Netzwerkmodus verwenden.
- `AwsvpcBlockIMDS`: Dieses optionale Flag blockiert den IMDS-Zugriff für die Task-Container, die im `awsvpc`-Netzwerkmodus ausgeführt werden.
- `AwsvpcAdditionalLocalRoutes`: Mit diesem optionalen Flag können Sie zusätzliche Routen im Aufgabennamespace haben.

  Ersetzen Sie `ip-address` mit der IP-Adresse für die zusätzlichen Routen, zum Beispiel 172.31.42.23/32.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster your_cluster_name -EnableTaskIAMRole -EnableTaskENI -
AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Bootstrapping von Amazon ECS-Windows-Container-Instances zur Datenübergabe

Wenn Sie eine Amazon EC2 EC2-Instance starten, können Sie Benutzerdaten an die EC2-Instance übergeben. Die Daten können für allgemeine automatisierte Konfigurationsaufgaben und sogar zur Ausführung von Skripts beim Start der Instance verwendet werden. In Amazon ECS werden Benutzerdaten hauptsächlich zur Übergabe von Konfigurationsinformationen an den Docker-Daemon und den Amazon-ECS-Container-Agenten verwendet.

Sie können mehrere Arten von Benutzerdaten an Amazon EC2 übergeben, darunter auch Cloud-Boothooks, Shell-Skripts und `cloud-init`-Anweisungen. Weitere Informationen zu diesen und anderen Formattypen finden Sie in der Cloud-Init-Dokumentation.

Sie können diese Benutzerdaten übergeben, wenn Sie den Amazon EC2-Start-Assistenten verwenden. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

Standardmäßige Windows-Benutzerdaten

Dieses Beispiel-Benutzerdatenskript zeigt die Standardbenutzerdaten, die Ihre Windows-Container-Instances erhalten, wenn Sie die Konsole verwenden. Das Skript unten führt Folgendes durch:

- Setzt den Clusternamen auf den Namen, den Sie eingegeben haben.
- Legt die IAM-Rollen für Aufgaben fest.
- Legt `json-file` und `awslogs` als verfügbare Protokolltreiber fest.

Darüber hinaus sind die folgenden Optionen verfügbar, wenn Sie den `awsvpc`-Netzwerkmodus verwenden.

- `EnableTaskENI`: Dieses Flag aktiviert das Aufgabennetzwerk und ist erforderlich, wenn Sie den `awsvpc`-Netzwerkmodus verwenden.
- `AwsvpcBlockIMDS`: Dieses optionale Flag blockiert den IMDS-Zugriff für die Task-Container, die im `awsvpc`-Netzwerkmodus ausgeführt werden.
- `AwsvpcAdditionalLocalRoutes`: Mit diesem optionalen Flag können Sie zusätzliche Routen haben.

  Ersetzen Sie `ip-address` mit der IP-Adresse für die zusätzlichen Routen, zum Beispiel 172.31.42.23/32.

Sie können dieses Skript für Ihre eigenen Container-Instances verwenden (vorausgesetzt, sie werden von dem für Amazon ECS optimierten Windows-Server-AMI gestartet).

Ersetzen Sie die Zeile `-Cluster` *`cluster-name`*, um Ihren eigenen Clusternamen anzugeben.

```
<powershell>
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]' -EnableTaskENI -AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Für Windows-Aufgaben, die für die Verwendung des `awslogs`-Protokolltreibers konfiguriert sind, müssen Sie auch die `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE`-Umgebungsvariable für die Container-Instance festlegen. Verwenden Sie die folgende Syntax:

Ersetzen Sie die Zeile `-Cluster` *`cluster-name`*, um Ihren eigenen Clusternamen anzugeben.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]'
</powershell>
```

Benutzerdaten für die Installation des Windows-Agenten

Mit diesem Beispiel-Benutzerdatenskript wird der Amazon-ECS-Container-Agent auf einer Instance installiert, die mit einem Windows_Server-2016-English-Full-Containers-AMI gestartet wurde. Es wurde anhand der Anweisungen zur Agenteninstallation auf der GitHubREADME-Seite des Amazon ECS Container Agent-Repositorys angepasst.

> **Note**
>
> Dieses Skript dient als Beispiel. Der Einstieg in Windows-Container ist mit dem Amazon-ECS-optimierten Windows Server-AMI wesentlich einfacher. Weitere Informationen finden Sie unter Erstellen eines Amazon ECS-Clusters für den Starttyp Fargate.

Sie können dieses Skript für Ihre eigenen Container-Instances verwenden (vorausgesetzt, sie werden mit einer Version des Windows_Server-2016-English-Full-Container-AMI gestartet). Achten Sie darauf, die Zeile *windows* zu ersetzen, um Ihren eigenen Cluster-Namen anzugeben (wenn Sie keinen Cluster namens `windows` verwenden).

```
<powershell>
# Set up directories the agent uses
New-Item -Type directory -Path ${env:ProgramFiles}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS\data -Force
# Set up configuration
$ecsExeDir = "${env:ProgramFiles}\Amazon\ECS"
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", "windows", "Machine")
[Environment]::SetEnvironmentVariable("ECS_LOGFILE", "${env:ProgramData}\Amazon\ECS\log
\ecs-agent.log", "Machine")
[Environment]::SetEnvironmentVariable("ECS_DATADIR", "${env:ProgramData}\Amazon\ECS
\data", "Machine")
# Download the agent
$agentVersion = "latest"
```

```
$agentZipUri = "https://s3.amazonaws.com/amazon-ecs-agent/ecs-agent-windows-
$agentVersion.zip"
$zipFile = "${env:TEMP}\ecs-agent.zip"
Invoke-RestMethod -OutFile $zipFile -Uri $agentZipUri
# Put the executables in the executable directory.
Expand-Archive -Path $zipFile -DestinationPath $ecsExeDir -Force
Set-Location ${ecsExeDir}
# Set $EnableTaskIAMRoles to $true to enable task IAM roles
# Note that enabling IAM roles will make port 80 unavailable for tasks.
[bool]$EnableTaskIAMRoles = $false
if (${EnableTaskIAMRoles}) {
  $HostSetupScript = Invoke-WebRequest https://raw.githubusercontent.com/aws/amazon-
ecs-agent/master/misc/windows-deploy/hostsetup.ps1
  Invoke-Expression $($HostSetupScript.Content)
}
# Install the agent service
New-Service -Name "AmazonECS" `
        -BinaryPathName "$ecsExeDir\amazon-ecs-agent.exe -windows-service" `
        -DisplayName "Amazon ECS" `
        -Description "Amazon ECS service runs the Amazon ECS agent" `
        -DependsOn Docker `
        -StartupType Manual
sc.exe failure AmazonECS reset=300 actions=restart/5000/restart/30000/restart/60000
sc.exe failureflag AmazonECS 1
Start-Service AmazonECS
</powershell>
```

Verwenden eines HTTP-Proxys für Amazon ECS Windows-Container-Instances

Sie können Ihre Amazon-ECS-Container-Instances für die Verwendung eines HTTP-Proxys sowohl für den Amazon-ECS-Container-Agenten als auch für den Docker-Daemon konfigurieren. Das ist praktisch, wenn Ihre Container-Instances keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance haben.

Um Ihre Amazon ECS Windows-Container-Instance für die Verwendung eines HTTP-Proxys zu konfigurieren, bestimmen Sie die folgenden Variablen beim Starten (mit Amazon EC2-Benutzerdaten).

`[Environment]::SetEnvironmentVariable("HTTP_PROXY", "http://proxy.mydomain:port", "Machine")`

Legen Sie `HTTP_PROXY` auf den Hostnamen (oder die IP-Adresse) und die Portnummer eines HTTP-Proxys fest, den der Amazon ECS-Agent verwenden soll, um eine Verbindung zum

Internet herzustellen. Beispielsweise haben Ihre Container-Instances vielleicht keinen externen Netzwerkzugriff über ein Amazon VPC-Internet-Gateway, NAT-Gateway oder eine Instance.

```
[Environment]::SetEnvironmentVariable("NO_PROXY",
"169.254.169.254,169.254.170.2,\\.\pipe\docker_engine", "Machine")
```

Setzen Sie `NO_PROXY` auf `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine`, um die EC2 Instance-Metadaten, IAM-Rollen für Aufgaben und Docker-Daemon-Datenverkehr aus dem Proxy zu filtern.

Example Windows HTTP-Proxy-Benutzerdatenskript

Das folgende PowerShell Beispielskript für Benutzerdaten konfiguriert den Amazon ECS-Container-Agenten und den Docker-Daemon so, dass sie einen von Ihnen angegebenen HTTP-Proxy verwenden. Sie können auch ein Cluster festlegen, in dem sich die Container-Instance selbst registriert.

Für die Verwendung dieses Skripts beim Starten einer Container-Instance befolgen Sie die Schritte in the section called “Starten einer Container-Instance” Kopieren Sie einfach das folgende PowerShell Skript und fügen Sie es in das Feld Benutzerdaten ein (achten Sie darauf, die roten Beispielwerte durch Ihre eigenen Proxy- und Clusterinformationen zu ersetzen).

> **Note**
>
> Die Option `-EnableTaskIAMRole` ist erforderlich, um IAM-Rollen für Aufgaben zu aktivieren. Weitere Informationen finden Sie unter Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance.

```
<powershell>
Import-Module ECSTools

$proxy = "http://proxy.mydomain:port"
[Environment]::SetEnvironmentVariable("HTTP_PROXY", $proxy, "Machine")
[Environment]::SetEnvironmentVariable("NO_PROXY", "169.254.169.254,169.254.170.2,\\.
\pipe\docker_engine", "Machine")

Restart-Service Docker
Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole
</powershell>
```

Konfiguration von Amazon ECS-Windows-Container-Instances für den Empfang von Spot-Instance-Benachrichtigungen

Amazon EC2 hält Ihre Spot-Instance an, beendet sie oder versetzt sie in den Ruhezustand, wenn der Spot-Preis den Höchstpreis für Ihre Anforderung überschreitet oder keine Kapazität mehr verfügbar ist. Amazon EC2 stellt eine Spot-Instance-Unterbrechungsbenachrichtigung bereit, was der Instance eine zweiminütige Warnung gibt, bevor sie unterbrochen wird. Wenn der Amazon-ECS-Spot-Instance-Ausgleich auf der Instance aktiviert ist, erhält ECS die Benachrichtigung über die Unterbrechung der Spot-Instance und versetzt die Instance in den Status `DRAINING`.

> **Important**
>
> Amazon ECS prüft auf Benachrichtigungen über die Unterbrechung der Spot-Instance, die über die Instance-Aktionen `terminate` und `stop` verfügen. Wenn Sie entweder das Verhalten `hibernate` für die Unterbrechung von Instances beim Anfordern Ihrer Spot-Instances oder die Spot-Flotte angegeben haben, wird der Amazon-ECS-Spot-Instance-Ausgleich für diese Instances nicht unterstützt.

Wenn eine Container-Instance auf `DRAINING` festgelegt wird, lässt es Amazon ECS nicht zu, dass die Platzierung neuer Aufgaben in der Container-Instance geplant wird. Serviceaufgaben auf der betroffenen Container-Instance mit dem Status `PENDING` werden umgehend gestoppt. Wenn Container-Instances im Cluster verfügbar sind, werden Ersatzserviceaufgaben darauf gestartet.

Sie können das Spot-Instance-Draining aktivieren, wenn Sie eine Instance starten. Sie müssen den `ECS_ENABLE_SPOT_INSTANCE_DRAINING`-Parameter festlegen, bevor Sie den Container-Agenten starten. Ersetzen Sie *`my-cluster`* durch den Namen Ihres Clusters.

```
[Environment]::SetEnvironmentVariable("ECS_ENABLE_SPOT_INSTANCE_DRAINING", "true",
 "Machine")

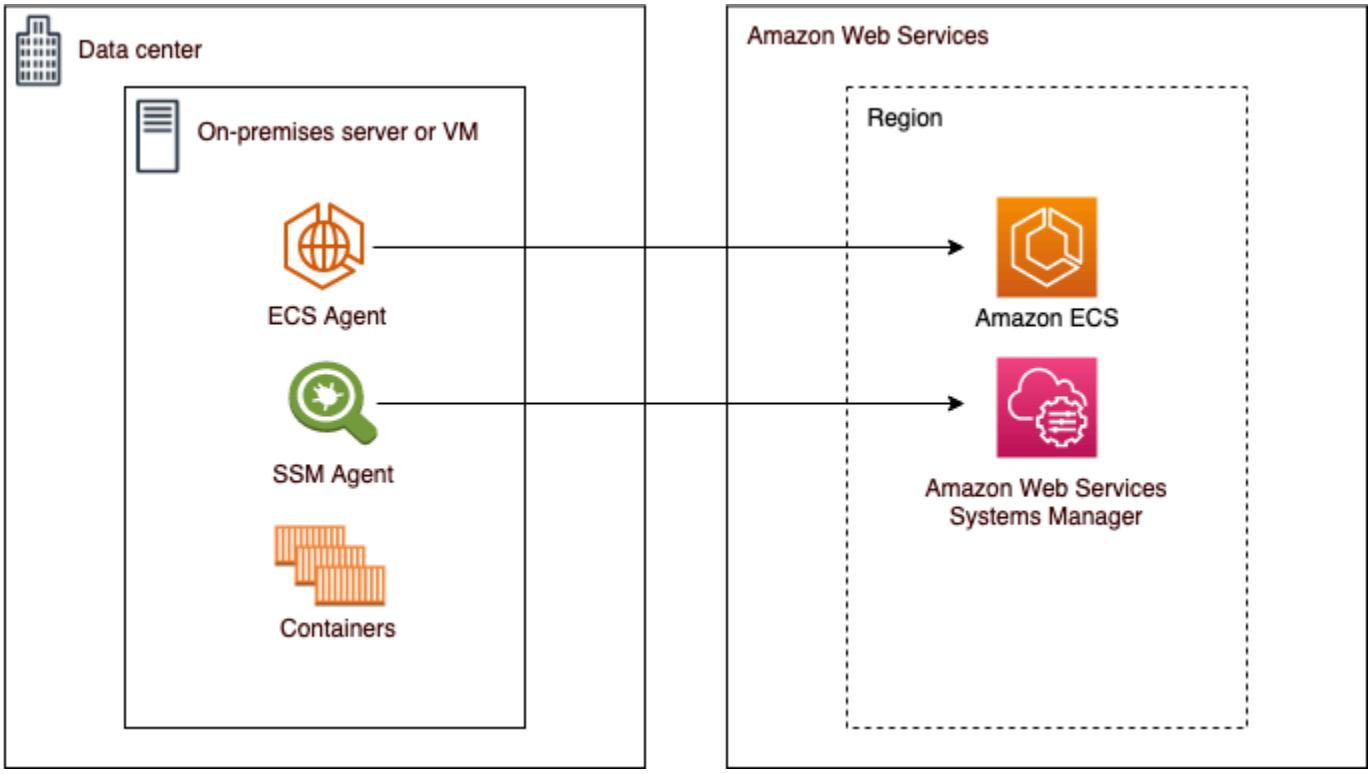
# Initialize the agent
Initialize-ECSAgent -Cluster my-cluster
```

Weitere Informationen finden Sie unter the section called “Starten einer Container-Instance”.

Amazon ECS-Cluster für den externen Starttyp

Amazon ECS Anywhere bietet Unterstützung für die Registrierung einer externen Instance, wie einem On-Premises-Server oder einer virtuellen Maschine (VM) in Ihren Amazon-ECS-Cluster. Externe Instances sind für ausgeführte Anwendungen optimiert, die ausgehenden Datenverkehr generieren oder Prozessdaten verarbeiten. Wenn Ihre Anwendung eingehenden Datenverkehr erfordert, wird die Ausführung dieser Arbeitslasten aufgrund fehlender Elastic Load Balancing-Unterstützung weniger effizient. Amazon ECS hat einen neuen `EXTERNAL`-Starttyp, mit dem Sie Dienste erstellen oder Tasks auf externen Instances ausführen können.

Nachfolgend finden Sie einen Überblick über die Systemarchitektur von Amazon ECS Anywhere. Auf Ihrem lokalen Server sind sowohl der Amazon ECS-Agent als auch der SSM-Agent installiert.



Unterstützte Betriebssysteme und Systemarchitekturen

Im Folgenden ist die Liste der unterstützten Betriebssysteme und Systemarchitekturen aufgeführt.

- Amazon Linux 2
- CentOS 7
- CentOS Stream 8

- RHEL 7, RHEL 8 – Weder Docker noch die offenen Paket-Repositories von RHEL unterstützen die native Installation von Docker auf RHEL. Sie müssen sicherstellen, dass Docker installiert ist, bevor Sie das Installationsskript ausführen, das in diesem Dokument beschrieben wird.
- Fedora 32, Fedora 33
- openSUSE Tumbleweed
- Ubuntu 18, Ubuntu 20, Ubuntu 22
- Debian 10

> **Important**
>
> Der Langzeitsupport (LTS-Support) für Debian 9 endete am 30. Juni 2022 und wird von Amazon ECS Anywhere nicht mehr unterstützt.

- Debian 11
- Debian 12 — Das NVIDIA Container-Toolkit wird derzeit auf Debian 12 nicht unterstützt. Sie werden keine GPUs auf Debian 12-Instanzen ausführen können.
- SUSE Enterprise Server 15
- Die `x86_64`- und `ARM64`-CPU-Architekturen werden unterstützt.
- Die folgenden Windows-Betriebssystemversionen werden unterstützt:
  - Windows Server 2022
  - Windows Server 2019
  - Windows Server 2016
  - Windows Server 20H2

Überlegungen

Seien Sie sich der folgenden Überlegungen bewusst, bevor Sie externe Instances verwenden.

- Sie können eine externe Instance auf einem Cluster gleichzeitig registrieren. Eine Anleitung zum Registrieren einer externen Instance bei einem anderen Cluster finden Sie unter Deregistrierung einer externen Amazon ECS-Instance.
- Ihre externen Instanzen benötigen eine IAM-Rolle, die es ihnen ermöglicht, mit AWS APIs zu kommunizieren. Weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere.

- Ihre externen Instances sollten keine vorkonfigurierte Instance-Anmeldeinformationskette lokal definiert haben, da dies das Registrierungsskript beeinträchtigt.
- Um Container-Logs an CloudWatch Logs zu senden, stellen Sie sicher, dass Sie in Ihrer Aufgabendefinition eine IAM-Rolle für die Aufgabenausführung erstellen und angeben.
- Wenn eine externe Instance in einem Cluster registriert ist, wird das `ecs.capability.external`-Attribut mit der Instance zugeordnet. Dieses Attribut identifiziert die Instance als externe Instance. Sie können Ihren externen Instances benutzerdefinierte Attribute hinzufügen, die als Einschränkung für die Platzierung von Vorgängen verwendet werden sollen. Weitere Informationen finden Sie unter Custom attributes (Benutzerdefinierte Attribute).
- Sie können Ihrer externen Instance Ressourcen-Tags hinzufügen. Weitere Informationen finden Sie unter Externe Container-Instances.
- ECS Exec wird für externe Instances unterstützt. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit ECS Exec.
- Im Folgenden finden Sie zusätzliche Überlegungen, die speziell für das Netzwerk mit Ihren externen Instances gelten. Weitere Informationen finden Sie unter Netzwerk .
  - Service-Load Balancing wird nicht unterstützt.
  - Service Discovery wird nicht unterstützt.
  - Tasks, die auf externen Instances ausgeführt werden, müssen die `bridge`, `host` oder `none`-Netzwerk-Modi verwenden. Der Netzwerkmodus `awsvpc` wird nicht unterstützt.
  - In jeder AWS Region gibt es Amazon ECS-Servicedomänen. Diese Service-Domains müssen den Datenverkehr an Ihre externen Instances senden dürfen.
  - Der auf Ihrer externen Instance installierte SSM Agent verwaltet IAM-Anmeldeinformationen, die alle 30 Minuten mit einem Hardware-Fingerabdruck gedreht werden. Wenn Ihre externe Instance die Verbindung zu verliert AWS, aktualisiert der SSM-Agent die Anmeldeinformationen automatisch, nachdem die Verbindung wiederhergestellt wurde. Weitere Informationen finden Sie unter Überprüfen von On-Premises-Servern und virtuellen Maschinen mit einem Hardware-Fingerprint im AWS Systems Manager -Benutzerhandbuch.
- Die `UpdateContainerAgent`-API wird nicht unterstützt. Anweisungen zum Aktualisieren des SSM Agent oder des Amazon-ECS-Agenten auf Ihren externen Instances finden Sie unter Aktualisierung des AWS Systems Manager Agenten und des Amazon ECS-Container-Agenten auf einer externen Instance.
- Amazon-ECS-Kapazitätsanbieter werden nicht unterstützt. Um einen Service zu erstellen oder einen eigenständigen Task für Ihre externen Instances auszuführen, verwenden Sie den `EXTERNAL`-Starttyp.

- SELinux wird nicht unterstützt.
- Verwenden von Amazon EFS Volumes oder Angeben eines `EFSVolumeConfiguration` wird nicht unterstützt.
- Die Integration mit App Mesh wird nicht unterstützt.
- Wenn Sie die Konsole verwenden, um eine externe Instance-Aufgabendefinition zu erstellen, müssen Sie die Aufgabendefinition mit dem JSON-Editor der Konsole erstellen.
- Wenn Sie ECS Anywhere auf Windows ausführen, müssen Sie Ihre eigene Windows-Lizenz in der On-Premises-Infrastruktur verwenden.
- Wenn Sie ein nicht für Amazon ECS optimiertes AMI verwenden, führen Sie die folgenden Befehle auf der externen Container-Instance aus, um Regeln für die Verwendung von IAM-Rollen für Aufgaben zu konfigurieren. Weitere Informationen finden Sie unter Zusätzliche Konfiguration der externen Instanz.

```
$ sysctl -w net.ipv4.conf.all.route_localnet=1
$ iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
$ iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Netzwerk

Externe Amazon-ECS-Instances sind für die Ausführung von Anwendungen optimiert, die ausgehenden Datenverkehr generieren oder Daten verarbeiten. Wenn Ihre Anwendung eingehenden Datenverkehr erfordert, z. B. einen Webdienst, macht das Fehlen von Elastic Load Balancing-Unterstützung die Ausführung dieser Workloads weniger effizient, da diese Workloads nicht hinter einem Load Balancer platziert werden können.

Im Folgenden finden Sie zusätzliche Überlegungen, die speziell für das Netzwerk mit Ihren externen Instances gelten.

- Service-Load Balancing wird nicht unterstützt.
- Service Discovery wird nicht unterstützt.
- Linux-Aufgaben, die auf externen Instances ausgeführt werden, müssen den Netzwerkmodus `bridge`, `host` oder `none` verwenden. Der Netzwerkmodus `awsvpc` wird nicht unterstützt.

  Weitere Informationen zu den jeweiligen Netzwerkmodi finden Sie unter Auswählen eines Netzwerkmodus im Leitfaden zu bewährten Methoden für Amazon ECS.

- Windows-Aufgaben, die auf externen Instances ausgeführt werden, müssen den `default`-Netzwerkmodus verwenden.
- Es gibt Amazon-ECS-Service-Domains in jeder Region, die den Datenverkehr an Ihre externen Instances senden dürfen.
- Der auf Ihrer externen Instance installierte SSM Agent verwaltet IAM-Anmeldeinformationen, die alle 30 Minuten mit einem Hardware-Fingerabdruck gedreht werden. Wenn Ihre externe Instanz die Verbindung zu verliert AWS, aktualisiert der SSM-Agent die Anmeldeinformationen automatisch, nachdem die Verbindung wiederhergestellt wurde. Weitere Informationen finden Sie unter Überprüfen von On-Premises-Servern und virtuellen Maschinen mit einem Hardware-Fingerprint im AWS Systems Manager -Benutzerhandbuch.

Die folgenden Domains werden für die Kommunikation zwischen dem Amazon-ECS-Service und dem Amazon-ECS-Agent verwendet, der auf Ihrer externen Instance installiert ist. Stellen Sie sicher, dass Datenverkehr zulässig ist und die DNS-Auflösung funktioniert. Für jeden Endpunkt repräsentiert *`Region`* die Regions-ID für eine von Amazon ECS unterstützte AWS -Region, z. B. `us-east-2` für die Region USA Ost (Ohio). Die Endpunkte für alle Regionen, die Sie verwenden, sollten zulässig sein. Für Endpunkte `ecs-a` und `ecs-t` sollten Sie ein Sternchen (z. B. `ecs-a-*`) einschließen.

- `ecs-a-*.`*`region`*`.amazonaws.com` – Dieser Endpunkt wird beim Verwalten von Aufgaben verwendet.
- `ecs-t-*.`*`region`*`.amazonaws.com` – Dieser Endpunkt wird zum Verwalten von Task- und Container-Metriken verwendet.
- `ecs.`*`region`*`.amazonaws.com` – Dies ist der Service-Endpunkt für Amazon ECS.
- `ssm.`*`region`*`.amazonaws.com` — Dies ist der Service-Endpunkt für. AWS Systems Manager
- `ec2messages.`*`region`*`.amazonaws.com`— Dies ist der Dienstendpunkt, der für die Kommunikation zwischen dem Systems Manager Manager-Agent und dem Systems Manager Manager-Dienst in der Cloud AWS Systems Manager verwendet wird.
- `ssmmessages.`*`region`*`.amazonaws.com`: Dieser Service-Endpunkt ist zum Erstellen und Löschen von Sitzungskanälen mit dem Session Manager-Service in der Cloud erforderlich..
- Wenn Ihre Aufgaben die Kommunikation mit anderen AWS Diensten erfordern, stellen Sie sicher, dass diese Dienstendpunkte zugelassen sind. Zu den Beispielanwendungen gehört die Verwendung von Amazon ECR zum Abrufen von Container-Images oder die Verwendung CloudWatch für CloudWatch Logs. Weitere Informationen finden Sie unter Service-Endpunkte in Allgemeine AWS -Referenz.

Amazon FSx for Windows File Server mit ECS Anywhere

Um die externen Instances Amazon FSx for Windows File Server mit Amazon ECS verwenden zu können, müssen Sie eine Verbindung zwischen Ihrem lokalen Rechenzentrum und dem AWS Cloud herstellen. Informationen zu den Optionen zum Verbinden Ihres Netzwerks mit Ihrer VPC finden Sie unter Verbindungsoptionen für Amazon Virtual Private Cloud.

gMSA mit ECS Anywhere

Die folgenden Anwendungsfälle werden für ECS Anywhere unterstützt.

- Das Active Directory befindet sich in der AWS Cloud — Für diese Konfiguration stellen Sie eine Verbindung zwischen Ihrem lokalen Netzwerk und der AWS Cloud Verwendung einer AWS Direct Connect Verbindung her. Informationen zum Erstellen der Verbindung finden Sie unter Konnektivitätsoptionen für Amazon Virtual Private Cloud. Sie erstellen ein Active Directory in der AWS Cloud. Informationen zu den ersten Schritten finden Sie AWS Directory Service unter Einrichtung AWS Directory Service im AWS Directory Service Administratorhandbuch. Anschließend können Sie Ihre externen Instanzen über die AWS Direct Connect Verbindung mit der Domain verbinden. Informationen zum Arbeiten mit gMSA mit Amazon ECS finden Sie unter the section called “Erfahren Sie, wie Sie gMSAs für EC2-Windows-Container verwenden”.
- Das Active Directory befindet sich im On-Premises-Rechenzentrum. – Für diese Konfiguration verbinden Sie Ihre externen Instances mit dem On-Premises-Active-Directory. Sie verwenden dann die lokal verfügbaren Anmeldeinformationen, wenn Sie die Amazon-ECS-Aufgaben ausführen.

Erstellen eines Amazon ECS-Clusters für den Starttyp Extern

Sie können mit der Amazon ECS-Konsole einen Amazon ECS-Cluster erstellen. Bevor Sie beginnen, vergewissern Sie sich, dass Sie die Schritte in Einrichtung für die Verwendung von Amazon ECS ausgeführt haben, und weisen Sie die entsprechende IAM-Berechtigung zu. Weitere Informationen finden Sie unter the section called “Beispiele für Amazon ECS-Cluster”. Die Amazon ECS-Konsole bietet eine einfache Möglichkeit, die Ressourcen zu erstellen, die von einem Amazon ECS-Cluster benötigt werden, indem ein AWS CloudFormation Stack erstellt wird.

Um den Prozess der Clustererstellung so einfach wie möglich zu gestalten, verfügt die Konsole über Standardauswahlen für viele Auswahlmöglichkeiten, die wir unten beschreiben. Es gibt auch Hilfe-Panels für die meisten Abschnitte in der Konsole, die weiteren Kontext bieten.

- Erstellt einen Standard-Namespace AWS Cloud Map , der denselben Namen wie der Cluster hat. Ein Namespace ermöglicht es Services, die Sie im Cluster erstellen, ohne zusätzliche Konfiguration eine Verbindung zu den anderen Services im Namespace herzustellen.

  Weitere Informationen finden Sie unter Amazon ECS-Services verbinden.

Sie können die folgenden Optionen ändern:

- Ändern Sie den Standard-Namespace, der dem Cluster zugeordnet ist.

  Über einen Namespace können Services, die Sie im Cluster erstellen, ohne zusätzliche Konfiguration eine Verbindung zu den anderen Services im Namespace herstellen. Der Standard-Namespace ist mit dem Cluster-Namen identisch. Weitere Informationen finden Sie unter Amazon ECS-Services verbinden.
- Den Cluster für externe Instances konfigurieren
- Aktivieren von Container Insights.

  CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Container Insights bietet auch Diagnoseinformationen, wie z. B. Fehler beim Container-Neustart, damit Sie Probleme schnell isolieren und beheben können. Weitere Informationen finden Sie unter the section called “Überwachen Sie Amazon ECS-Container mit Container Insights”.
- Fügen Sie Tags hinzu, die Ihnen helfen, Ihren Cluster zu identifizieren.

So erstellen Sie einen neuen Cluster (Amazon ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Clusters die Option Create cluster (Cluster erstellen) aus.
5. Konfigurieren Sie unter Clusterkonfiguration Folgendes:

   - Geben Sie für Cluster-Name einen eindeutigen Namen ein.

     Der Name kann bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern und Bindestriche enthalten.

- (Optional) Wenn der für Service Connect verwendete Namespace nicht mit dem Clusternamen identisch sein soll, geben Sie unter Namespace einen eindeutigen Namen ein.

6. Erweitern Sie Infrastruktur und wählen Sie AWS Fargate (serverlos) aus.
7. (Optional) Um Container Insights zu aktivieren, erweitern Sie Monitoring (Überwachung) und aktivieren Sie dann Use Container Insights (Container Insights verwenden).
8. (Optional) Um Ihren Cluster leichter identifizieren zu können, erweitern Sie den Bereich Tags, und konfigurieren Sie dann Ihre Tags.

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
9. Wählen Sie Erstellen.

Nächste Schritte

Sie müssen die Instanzen beim Cluster registrieren. Weitere Informationen finden Sie unter Registrierung einer externen Instance in einem Amazon ECS-Cluster.

Nachdem Sie den Cluster erstellt haben, können Sie Aufgabendefinitionen für Ihre Anwendungen erstellen und diese dann als eigenständige Aufgaben oder als Teil eines Services ausführen. Weitere Informationen finden Sie hier:

- Amazon-ECS-Aufgabendefinitionen
- Eine Anwendung als Amazon ECS-Aufgabe ausführen
- Einen Amazon ECS-Service mithilfe der Konsole erstellen

Registrierung einer externen Instance in einem Amazon ECS-Cluster

Für jede externe Instance, die Sie bei einem Amazon-ECS-Cluster registrieren, muss der SSM Agent, der Amazon ECS Container Agent und Docker installiert sein. Um die externe Instance in einem Amazon ECS-Cluster zu registrieren, muss sie zunächst als AWS Systems Manager verwaltete Instance registriert werden. Sie können das Installationsskript mit wenigen Klicks auf der Amazon-ECS-Konsole erstellen. Das Installationsskript enthält einen Systems Manager Aktivierungsschlüssel und Befehle zur Installation aller erforderlichen Agents und Docker. Das

Installationsskript muss auf Ihrem On-Premises-Server oder VM ausgeführt werden, um die Installations- und Registrierungsschritte abzuschließen.

> **Note**
>
> Bevor Sie Ihre externe Linux-Instance beim Cluster anmelden, erstellen Sie die `/etc/ecs/ecs.config`-Datei auf Ihrer externen Instance und fügen alle gewünschten Container-Agent-Konfigurationsparameter hinzu. Sie können dies nicht tun, nachdem Sie die externe Instance in einem Cluster registriert haben. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Wählen Sie auf der Seite Cluster einen Cluster aus, bei dem Ihre externe Instance registriert werden soll.
5. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur).
6. Führen Sie auf der Seite Register external instances (Externe Instances anmelden) die folgenden Schritte aus.

    a. Geben sie für Dauer des Aktivierungsschlüssels (in Tagen) die Anzahl der Tage ein, für die der Aktivierungsschlüssel aktiv bleibt. Nach der Anzahl der Tage, die Sie eingegeben haben, funktioniert der Schlüssel nicht mehr bei der Registrierung einer externen Instance.

    b. Geben Sie für die Anzahl der Instances die Anzahl der externen Instances ein, die Sie mit dem Aktivierungsschlüssel bei Ihrem Cluster registrieren möchten.

    c. Wählen Sie für Instance-Rolle die IAM-Rolle aus, die Sie Ihren externen Instances zuordnen möchten. Wenn noch keine Rolle erstellt wurde, wählen Sie Erstellen einer neuen Rolle, damit Sie Amazon ECS eine Rolle in Ihrem Namen erstellt. Weitere Informationen zu den erforderlichen IAM-Berechtigungen für Ihre externen Instances finden Sie unter IAM-Rolle in Amazon ECS Anywhere.

    d. Kopieren Sie den Anmeldungsbefehl. Dieser Befehl sollte auf jeder externen Instance ausgeführt werden, die Sie beim Cluster registrieren möchten.

> **Important**
>
> Der Bash-Teil des Skripts muss als Root ausgeführt werden. Wenn der Befehl nicht als Root ausgeführt wird, wird ein Fehler zurückgegeben.

e. Klicken Sie auf Schließen.

AWS CLI for Linux operating systems

1. Erstellen Sie ein Systems Manager-Aktivierungspaar. Dies wird für die Aktivierung verwalteter Instances von Systems Manager verwendet. Die Ausgabe enthält eine `ActivationId` und `ActivationCode`. Sie werden sie in einem späteren Schritt verwenden. Stellen Sie sicher, dass Sie die ECS Anywhere IAM-Rolle angeben, die Sie erstellt haben. Weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Laden Sie das Installationsskript auf dem On-Premises-Server oder der virtuellen Maschine (VM) herunter.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"
```

3. (Optional) Führen Sie auf Ihrem On-Premises-Server oder Ihrer virtuellen Maschine (VM) die folgenden Schritte aus, um das Installationsskript mithilfe der Skriptsignaturdatei zu überprüfen.

   a. Laden Sie GnuPG herunter und installieren Sie es. Weitere Informationen zu GNUpg finden Sie auf der GnuPG-Website. Für Linux-Systeme installieren Sie `gpg` mit dem Paketmanager auf Ihrer Linux-Variante.

   b. Rufen Sie den öffentlichen Amazon-ECS-PGP-Schlüssel ab.

```
gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
```

   c. Laden Sie die Signatur des Installationsskripts herunter. Die Signaturen sind ASCII-getrennte PGP-Signaturen, die in Dateien mit der Erweiterung `.asc` gespeichert sind.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh.asc" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh.asc"
```

d. Überprüfen Sie die Installationsskriptdatei mit dem Schlüssel.

```
gpg --verify /tmp/ecs-anywhere-install.sh.asc /tmp/ecs-anywhere-install.sh
```

Folgendes ist die erwartete Ausgabe:

```
gpg: Signature made Tue 25 May 2021 07:16:29 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@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: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Führen Sie auf dem On-Premises-Server oder der virtuellen Maschine (VM) das Installationsskript aus. Geben Sie im ersten Schritt den Clusternamen, die Region und die Aktivierungs-ID des Systems Manager sowie den Aktivierungscode an.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE
```

Für einen On-Premises-Server oder eine virtuelle Maschine (VM), auf der der NVIDIA-Treiber für GPU-Workloads installiert ist, müssen Sie das `--enable-gpu`-Flag zum Installationsskript hinzufügen. Wenn dieses Flag angegeben wird, überprüft das Installationsskript, ob der NVIDIA-Treiber ausgeführt wird, und fügt dann die erforderlichen Konfigurationsvariablen zum Ausführen Ihrer Amazon-ECS-Aufgaben hinzu. Weitere Informationen zum Ausführen von GPU-Workloads und zum Angeben von GPU-Anforderungen in einer Aufgabendefinition finden Sie unter Angeben von GPUs in einer Amazon ECS-Aufgabendefinition.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
```

```
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE \
    --enable-gpu
```

Führen Sie die folgenden Schritte aus, um eine vorhandene externe Instance bei einem anderen Cluster zu registrieren.

So registrieren Sie eine vorhandene externe Instance in einem anderen Cluster

1. Halten Sie den Amazon-ECS-Container-Agent an.

```
sudo systemctl stop ecs.service
```

2. Bearbeiten Sie die `/etc/ecs/ecs.config`-Datei und stellen Sie auf der `ECS_CLUSTER`-Zeile sicher, dass der Clustername mit dem Namen des Clusters übereinstimmt, bei dem die externe Instance registriert werden soll.
3. Entfernen Sie die vorhandenen Amazon-ECS-Agentendaten.

```
sudo rm /var/lib/ecs/data/agent.db
```

4. Starten Sie den Amazon-ECS-Container-Agenten.

```
sudo systemctl start ecs.service
```

AWS CLI for Windows operating systems

1. Erstellen Sie ein Systems Manager-Aktivierungspaar. Dies wird für die Aktivierung verwalteter Instances von Systems Manager verwendet. Die Ausgabe enthält eine `ActivationId` und `ActivationCode`. Sie werden sie in einem späteren Schritt verwenden. Stellen Sie sicher, dass Sie die ECS Anywhere IAM-Rolle angeben, die Sie erstellt haben. Weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Laden Sie das Installationsskript auf dem On-Premises-Server oder der virtuellen Maschine (VM) herunter.

```
Invoke-RestMethod -URI "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install.ps1" -OutFile “ecs-anywhere-install.ps1”
```

3. (Optional) Das Powershell-Skript wird von Amazon signiert und daher führt Windows die Zertifikatvalidierung automatisch auf derselben Weise durch. Sie müssen keine manuelle Validierung durchführen.

   Um das Zertifikat manuell zu überprüfen, klicken Sie mit der rechten Maustaste auf die Datei, navigieren Sie zu Eigenschaften und verwenden Sie die Registerkarte Digitale Signaturen, um weitere Details zu erhalten.

   Diese Option ist nur verfügbar, wenn der Host das Zertifikat im Zertifikatspeicher hat.

   Die Verifizierung sollte in etwa folgendermaßen aussehen:

```
# Verification (PowerShell)
Get-AuthenticodeSignature -FilePath .\ecs-anywhere-install.ps1

SignerCertificate                         Status      Path
-----------------                         ------      ----
EXAMPLECERTIFICATE                        Valid       ecs-anywhere-install.ps1

...

Subject              : CN="Amazon Web Services, Inc.",...

----
```

4. Führen Sie auf dem On-Premises-Server oder der virtuellen Maschine (VM) das Installationsskript aus. Geben Sie im ersten Schritt den Clusternamen, die Region und die Aktivierungs-ID des Systems Manager sowie den Aktivierungscode an.

```
.\ecs-anywhere-install.ps1 -Region $Region -Cluster $Cluster -
ActivationID $ActivationID -ActivationCode $ActivationCode
```

5. Stellen Sie sicher, dass der Amazon-ECS-Container-Agent ausgeführt wird.

```
Get-Service AmazonECS

Status   Name               DisplayName
------   ----               -----------
```

```
Running  AmazonECS          Amazon ECS
```

Führen Sie die folgenden Schritte aus, um eine vorhandene externe Instance bei einem anderen Cluster zu registrieren.

So registrieren Sie eine vorhandene externe Instance in einem anderen Cluster

1. Halten Sie den Amazon-ECS-Container-Agent an.

   ```
   Stop-Service AmazonECS
   ```

2. Ändern Sie den Parameter ECS_CLUSTER so, dass der Clustername mit dem Namen des Clusters übereinstimmt, bei dem die externe Instance angemeldet werden soll.

   ```
   [Environment]::SetEnvironmentVariable("ECS_CLUSTER", $ECSCluster,
    [System.EnvironmentVariableTarget]::Machine)
   ```

3. Entfernen Sie die vorhandenen Amazon-ECS-Agentendaten.

   ```
   Remove-Item -Recurse -Force $env:ProgramData\Amazon\ECS\data\*
   ```

4. Starten Sie den Amazon-ECS-Container-Agenten.

   ```
   Start-Service AmazonECS
   ```

Das AWS CLI kann verwendet werden, um eine Systems Manager Manager-Aktivierung zu erstellen, bevor das Installationsskript ausgeführt wird, um den Registrierungsprozess für externe Instanzen abzuschließen.

Deregistrierung einer externen Amazon ECS-Instance

Wir empfehlen, dass Sie die Instance sowohl bei Amazon ECS als auch AWS Systems Manager danach abmelden, wenn Sie mit der Instance fertig sind. Nach Aufhebung der Registrierung kann die externe Instance keine neuen Aufgaben mehr akzeptieren.

Wenn zum Zeitpunkt der Abmeldung auf der Container-Instance Aufgaben ausgeführt werden, bleiben diese Aufgaben aktiv, bis die Aufgaben auf andere Weise gestoppt werden. Diese Aufgaben werden jedoch nicht mehr von Amazon ECS überwacht bzw. erfasst. Falls diese Aufgaben auf Ihrer

externen Instance Teil eines Amazon-ECS-Service ist, startet der Service-Scheduler eine andere Kopie der betreffenden Aufgabe in einer anderen Instance, sofern dies möglich ist.

Nachdem Sie die Instance deregistriert haben, bereinigen Sie die verbleibenden AWS Ressourcen auf der Instance. Sie können sie dann in einem neuen Cluster registrieren.

Verfahren

AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre externe Instance registriert ist.
3. Wählen Sie im Navigationsbereich Clusters und danach den Cluster aus, der Ihre externe Instance hostet.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur).
5. Wählen Sie unter Container instances (Container-Instances) die externe Instance-ID aus, deren Anmeldung aufgehoben werden soll. Sie werden zur Detailseite für Container-Instance umgeleitet.
6. Wählen Sie auf der Seite Container Instance : ***id*** Deregister aus.
7. Überprüfen Sie die Meldung zum Abmelden. Wählen Sie Abmelden von AWS Systems Manager aus, um die externe Instance auch als verwaltete Instance von Systems Manager abzumelden. Wählen Sie Deregister.

   > **Note**
   >
   > Sie können die externe Instance in der Systems Manager Konsole als verwaltete Instance von Systems Manager abmelden. Anweisungen finden Sie unter Abmelden von verwalteten Instances im AWS Systems Manager -Benutzerhandbuch.

8. Nachdem Sie die Registrierung der Instanz aufgehoben haben, bereinigen Sie die AWS Ressourcen auf Ihrem lokalen Server oder Ihrer VM.

Betriebssystem	Schritte	
Linux	a. Beenden Sie den Amazon-ECS-Contain	

Betriebssystem	Schritte	

	er-Agent und die SSM Agent-Dienste auf der Instance.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Entfernen Sie die Amazon-ECS- und Systems Manager Pakete.
	Für CentOS 7, CentOS 8 und RHEL 7
	`sudo yum remove -y amazon-ecs-init amazon-ssm-agent`
	Für SUSE Enterprise Server 15
	`sudo zypper remove -y amazon-ecs-init amazon-ssm-agent`
	Für Debian und Ubuntu
	`sudo apt remove -y amazon-ecs-init amazon-ssm-agent`
	c. Entfernen Sie die übrig gebliebenen Verzeichnisse.	

Betriebssystem	Schritte
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`

Windows	a. Beenden Sie den Amazon-ECS-Container-Agent und die SSM Agent-Dienste auf der Instance.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Entfernen Sie das Amazon-ECS-Paket.
	`.\ecs-anywhere-install.ps1 -Uninstall`

AWS CLI

1. Sie benötigen die Instance-ID und den Container-Instance-ARN, um die Registrierung der Container-Instance aufzuheben. Wenn Sie diese Werte nicht haben, führen Sie die folgenden Befehle aus

   Führen Sie den folgenden Befehl aus, um die Instance-ID zu erhalten.

   Sie verwenden die Instance-ID (`instanceID`), um den Container-Instance ARN (`containerInstanceARN`) abzurufen.

```
instanceId=$(aws ssm describe-instance-information --region "{{ region }}" |
 jq ".InstanceInformationList[] |select(.IPAddress==\"{{ IPv4 Address }}\")
 | .InstanceId" | tr -d'"'
```

Führen Sie die folgenden Befehle aus.

Sie verwenden `containerInstanceArn` als Parameter im Befehl, um die Registrierung der Instance (`deregister-container-instance`) aufzuheben.

```
instances=$(aws ecs list-container-instances --cluster "{{ cluster }}" --region
 "{{ region }}" | jq -c '.containerInstanceArns')
containerInstanceArn=$(aws ecs describe-container-instances --cluster
 "{{ cluster }}" --region "{{ region }}" --container-instances $instances
 | jq ".containerInstances[] | select(.ec2InstanceId==\"{{ instanceId }}\")
 | .containerInstanceArn" | tr -d '"')
```

2. Führen Sie den folgenden Befehl aus, um die Instance zu leeren.

```
aws ecs update-container-instances-state --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instances "{{ containerInstanceArn }}" --status
 DRAINING
```

3. Nachdem die Container-Instance den Ausgleich abgeschlossen hat, führen Sie den folgenden Befehl aus, um die Registrierung der Instance aufzuheben.

```
aws ecs deregister-container-instance --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instance "{{ containerInstanceArn }}"
```

4. Führen Sie den folgenden Befehl aus, um die Container-Instance aus SSM zu entfernen.

```
aws ssm deregister-managed-instance --region "{{ region }}" --instance-id
 "{{ instanceId }}"
```

5. Nachdem Sie die Registrierung der Instanz aufgehoben haben, bereinigen Sie die AWS Ressourcen auf Ihrem lokalen Server oder Ihrer VM.

Betriebssystem	Schritte	
Linux	a. Beenden Sie den Amazon-ECS-Contain	

Betriebssystem	Schritte	

	er-Agent und die SSM Agent-Dienste auf der Instance.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Entfernen Sie die Amazon-ECS- und Systems Manager Pakete.
	`sudo (yum/apt/zypper) remove amazon-ecs-init amazon-ssm-agent`
	c. Entfernen Sie die übrig gebliebenen Verzeichnisse.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Betriebssystem	Schritte

Windows	a. Beenden Sie den Amazon-ECS-Container-Agent und die SSM Agent-Dienste auf der Instance.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Entfernen Sie das Amazon-ECS-Paket.
	`.\ecs-anywhere-install.ps1 -Uninstall`

Aktualisierung des AWS Systems Manager Agenten und des Amazon ECS-Container-Agenten auf einer externen Instance

Auf Ihrem lokalen Server oder Ihrer VM müssen sowohl der AWS Systems Manager Agent (SSM Agent) als auch der Amazon ECS-Container-Agent ausgeführt werden, wenn Amazon ECS-Workloads ausgeführt werden. AWS veröffentlicht neue Versionen dieser Agenten, wenn Funktionen hinzugefügt oder aktualisiert werden. Wenn Ihre externen Instances eine frühere Version eines Agent verwenden, können Sie diese mithilfe der folgenden Verfahren aktualisieren.

Aktualisieren des SSM Agent auf einer externen Instance

AWS Systems Manager empfiehlt, dass Sie den Aktualisierungsprozess des SSM-Agenten auf Ihren Instances automatisieren. Sie bieten verschiedene Methoden zur Automatisierung von Updates. Weitere Informationen finden Sie unter Automatisieren von Updates für SSM Agent im AWS Systems Manager -Benutzerhandbuch.

Aktualisieren des Amazon-ECS-Agenten auf einer externen Instance

Auf Ihren externen Instances wird der Amazon-ECS-Container-Agent aktualisiert, indem das `ecs-init`-Paket aktualisiert wird. Die Aktualisierung des Amazon-ECS-Agenten unterbricht nicht die ausgeführten Tasks oder Services. Amazon ECS stellt das `ecs-init`-Paket und Signaturdatei in einem Amazon S3 Bucket in jeder Region. Beginnend mit `ecs-init`-Version `1.52.1-1`, bietet Amazon ECS separate `ecs-init`-Pakete für die Verwendung in Abhängigkeit vom Betriebssystem und der Systemarchitektur an, die Ihre externe Instance verwendet.

Verwenden Sie die folgende Tabelle, um das `ecs-init`-Paket zu bestimmen, das Sie basierend auf dem Betriebssystem und der Systemarchitektur, die Ihre externe Instance verwendet, herunterladen sollten.

> **Note**
>
> Mit den folgenden Befehlen können Sie bestimmen, welches Betriebssystem und welche Systemarchitektur Ihre externe Instance verwendet.
>
> ```
> cat /etc/os-release
> uname -m
> ```

Betriebssysteme (Architektur)	ecs-init-Paket
CentOS 7 (x86_64) CentOS 8 (x86_64) SUSE Enterprise Server 15 (x86_64) RHEL 7 (x86_64) RHEL 8 (x86_64)	`amazon-ecs-init-latest.x86_64.rpm`
CentOS 7 (aarch64) CentOS 8 (aarch64) RHEL 7 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`

Betriebssysteme (Architektur)	ecs-init-Paket
Debian 9 (x86_64) Debian 10 (x86_64) Debian 11 (x86_64) Debian 12 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64)	`amazon-ecs-init-latest.amd64.deb`
Debian 9 (aarch64) Debian 10 (aarch64) Debian 11 (aarch64) Debian 12 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (aarch64)	`amazon-ecs-init-latest.arm64.deb`

Gehen Sie folgendermaßen vor, um den Amazon-ECS-Agenten zu aktualisieren.

So aktualisieren Sie den Amazon-ECS-Agenten

1. Bestätigen Sie die Version des Amazon-ECS-Agenten, die Sie ausführen.

   ```
   curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
   ```

2. Herunterladen des `ecs-init`-Paket für Ihr Betriebssystem und Systemarchitektur. Amazon ECS stellt die `ecs-init`-Paketdatei in einem Amazon S3 Bucket in jeder Region. Stellen Sie sicher, dass Sie den *`<region>`*-Bezeichner im Befehl mit dem Regionsnamen (z. B. `us-west-2`), dem Sie geografisch am nächsten sind, ersetzen.

   amazon-ecs-init-latest.x86_64.rpm

```
curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm
```

amazon-ecs-init-latest.aarch64.rpm

```
curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb
```

3. (Optional) Überprüfen der Gültigkeit der `ecs-init`-Paketdatei mit der PGP-Signatur.

   a. Laden Sie GnuPG herunter und installieren Sie es. Weitere Informationen zu GNUpg finden Sie auf der GnuPG-Website. Für Linux-Systeme installieren Sie gpg mit dem Paketmanager auf Ihrer Linux-Variante.

   b. Rufen Sie den öffentlichen Amazon-ECS-PGP-Schlüssel ab.

   ```
   gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
   ```

   c. Laden Sie die `ecs-init`-Paket-Signaturdatei herunter. Die Signatur ist eine ASCII-getrennte PGP-Signatur, die in einer Datei mit der Erweiterung `.asc` gespeichert ist. Amazon ECS stellt die Signaturdatei in einem Amazon S3 Bucket in jeder Region bereit. Stellen Sie sicher, dass Sie den *<region>*-Bezeichner im Befehl mit dem Regionsnamen (z. B. `us-west-2`), dem Sie geografisch am nächsten sind, ersetzen.

   amazon-ecs-init-latest.x86_64.rpm

   ```
   curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm.asc
   ```

amazon-ecs-init-latest.aarch64.rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm.asc
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb.asc
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb.asc
```

d. Verifizieren der `ecs-init`-Paketdatei mit dem Schlüssel.

Für die **`rpm`**-Pakete

```
gpg --verify amazon-ecs-init.rpm.asc ./amazon-ecs-init.rpm
```

Für die **`deb`**-Pakete

```
gpg --verify amazon-ecs-init.deb.asc ./amazon-ecs-init.deb
```

Folgendes ist die erwartete Ausgabe:

```
gpg: Signature made Fri 14 May 2021 09:31:36 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@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: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Installieren Sie das Paket `ecs-init`.

Für das **`rpm`**-Paket auf CentOS 7, CentOS 8 und RHEL 7

```
sudo yum install -y ./amazon-ecs-init.rpm
```

Für das **rpm**-Paket auf SUSE Enterprise Server 15

```
sudo zypper install -y --allow-unsigned-rpm ./amazon-ecs-init.rpm
```

Für das **deb**-Paket

```
sudo dpkg -i ./amazon-ecs-init.deb
```

5. Den Service `ecs` neu starten.

```
sudo systemctl restart ecs
```

6. Überprüfen Sie, ob die Amazon-ECS-Agent-Version aktualisiert wurde.

```
curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

Aktualisierung eines Amazon ECS-Clusters

Sie können die folgenden Cluster-Eigenschaften ändern:

- Einen Standard-Kapazitätsanbieter festlegen

  Jeder Cluster kann über einen oder mehrere Kapazitätsanbieter und eine optionale Kapazitätsanbieter-Standardstrategie verfügen. Die Kapazitätsanbieterstrategie legt fest, wie die Aufgaben über die Kapazitätsanbieter eines Clusters verteilt werden. Wenn Sie eine eigenständige Aufgabe ausführen oder einen Service erstellen, können Sie entweder die Kapazitätsanbieter-Standardstrategie des Clusters verwenden oder eine Strategie für Kapazitätsanbieter angeben, die die Standardstrategie überschreibt.
- Aktivieren von Container Insights.

  CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen. Container Insights bietet auch Diagnoseinformationen, wie z. B. Fehler beim Container-Neustart, damit Sie Probleme schnell isolieren und beheben können. Weitere Informationen finden Sie unter the section called “Überwachen Sie Amazon ECS-Container mit Container Insights”.

- Fügen Sie Tags hinzu, um Ihnen bei der Identifizierung Ihrer Cluster zu helfen.

Verfahren

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster: ***name*** Cluster aktualisieren.
5. Um den Standardkapazitätsanbieter festzulegen, wählen Sie unter Standardstrategie für Kapazitätsanbieter die Option Weitere hinzufügen aus.

   a. Wählen Sie für Kapazitätsanbieter den Kapazitätsanbieter aus.

   b. (Optional) Geben Sie für Basis die Mindestanzahl von Aufgaben ein, die auf dem Kapazitätsanbieter ausgeführt werden.

      Sie können nur einen Basis-Wert für einen Kapazitätsanbieter festlegen.

   c. (Optional) Geben Sie für Gewicht den relativen Prozentsatz der Gesamtzahl der gestarteten Aufgaben ein, die den angegebenen Kapazitätsanbieter nutzen.

   d. (Optional) Wiederholen Sie die Schritte für alle weiteren Kapazitätsanbieter.
6. Um Container Insights zu aktivieren oder zu deaktivieren, erweitern Sie Überwachen und aktivieren Sie dann Container-Insights verwenden.
7. Um Ihren Cluster leichter identifizieren zu können, erweitern Sie den Bereich Tags, und konfigurieren Sie dann Ihre Tags.

   [Markierung hinzufügen] Wählen Sie Add tag (Markierung hinzufügen), und führen Sie die folgenden Schritte aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.

   [Tag (Markierung) entfernen] Wählen Sie Remove (Entfernen) rechts neben dem Schlüssel und dem Wert des Tags (Markierung).
8. Wählen Sie Aktualisieren.

Löschen eines Amazon ECS-Clusters

Wenn Sie ein Cluster nicht mehr verwenden, können Sie es löschen. Nachdem Sie den Cluster gelöscht haben, wechselt er zum `INACTIVE`-Zustand. Cluster mit dem Status `INACTIVE` Status können für einen bestimmten Zeitraum im Konto erkennbar bleiben. Dieses Verhalten kann sich jedoch in Zukunft ändern und Sie sollten sich nicht darauf verlassen, dass `INACTIVE`-Cluster bestehen bleiben.

Bevor Sie einen Cluster löschen, müssen Sie die folgenden Vorgänge ausführen:

- Löschen aller Services im Cluster. Weitere Informationen finden Sie unter the section called “Löschen eines Service”.
- Anhalten aller derzeit ausgeführten Aufgaben. Weitere Informationen finden Sie unter the section called “Beenden einer Aufgabe”.
- Abmelden aller angemeldeten Container-Instances im Cluster. Weitere Informationen finden Sie unter the section called “Aufheben der Registrierung einer Container-Instance”.
- Löschen Sie den -Namespace. Weitere Informationen finden Sie unter Löschen von Namespaces im AWS Cloud Map Entwicklerleitfaden.

Verfahren

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
3. Klicken Sie im Navigationsbereich auf Cluster.
4. Klicken Sie auf der Seite Clusters (Cluster) auf die zu löschenden Cluster.
5. Wählen Sie oben rechts auf der Seite Delete Cluster (Cluster löschen) aus.

   Es wird eine Meldung angezeigt, wenn Sie nicht alle dem Cluster zugeordneten Ressourcen gelöscht haben.
6. Geben Sie im Bestätigungsfeld delete ***`Clustername`*** ein.

Einen Kapazitätsanbieter für Amazon ECS erstellen

Nach Abschluss der Cluster-Erstellung können Sie einen neuen Kapazitätsanbieter (Auto-Scaling-Gruppe) für den EC2-Starttyp erstellen.

Vor dem Erstellen des Kapazitätsanbieters müssen Sie eine Auto-Scaling-Gruppe erstellen. Weitere Informationen zu Auto-Scaling-Gruppen finden Sie im Benutzerhandbuch für Amazon EC2 Auto Scaling.

So erstellen Sie einen Kapazitätsanbieter für den Cluster (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster: ***Name*** die Option Infrastructure (Infrastruktur) und dann Create (Erstellen) aus.
5. Konfigurieren Sie auf der Seite Create capacity providers (Kapazitätsanbieter erstellen) die folgenden Optionen.

    a. Geben Sie unter Basic details (Allgemeine Details) für Capacity provider name (Name des Kapazitätsanbieters), einen eindeutigen Namen des Kapazitätsanbieters ein.

    b. Wählen Sie unter Auto Scaling group (Auto-Scaling-Gruppe) für Use an existing Auto Scaling group (Vorhandene Auto-Scaling-Gruppe verwenden) die Auto-Scaling-Gruppe aus.

    c. (Optional) Um eine Skalierungsrichtlinie zu konfigurieren, konfigurieren Sie unter Scaling policies (Skalierungsrichtlinien) die folgenden Optionen.

    - Damit Amazon ECS die Ab- und Aufskalierungsaktionen verwaltet, wählen Sie Turn on managed scaling (Verwaltete Skalierung aktivieren) aus.
    - Um zu verhindern, dass eine EC2-Instance mit ausgeführten Amazon-ECS-Aufgaben beendet wird, wählen Sie Turn on scaling protection (Skalierungsschutz aktivieren) aus.
    - Geben Sie unter Zielkapazität festlegen den Zielwert für die CloudWatch Metrik ein, die in der von Amazon ECS verwalteten Skalierungsrichtlinie für die Zielverfolgung verwendet wird.

6. Wählen Sie Erstellen.

Aktualisierung eines Amazon ECS-Kapazitätsanbieters

Wenn Sie eine Auto-Scaling-Gruppe als Kapazitätsanbieter verwenden, können Sie die Skalierungsrichtlinie der Gruppe ändern.

Um einen Kapazitätsanbieter für den Cluster zu aktualisieren (Amazon ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster: ***Name*** die Option Infrastructure (Infrastruktur) und dann Update (Aktualisieren) aus.
5. Konfigurieren Sie auf der Seite Create capacity providers (Kapazitätsanbieter erstellen) die folgenden Optionen.

    - Konfigurieren Sie in der Auto-Scaling-Gruppe unter Scaling-Richtlinien die folgenden Optionen.

        - Damit Amazon ECS die Ab- und Aufskalierungsaktionen verwaltet, wählen Sie Turn on managed scaling (Verwaltete Skalierung aktivieren) aus.
        - Um zu verhindern, dass EC2-Instances mit ausgeführten Amazon-ECS-Aufgaben beendet werden, wählen Sie Skalierungsschutz aktivieren aus.
        - Geben Sie unter Zielkapazität festlegen den Zielwert für die CloudWatch Metrik ein, die in der von Amazon ECS verwalteten Skalierungsrichtlinie für die Zielverfolgung verwendet wird.
6. Wählen Sie Aktualisieren.

Löschen eines Amazon ECS-Kapazitätsanbieters

Wenn Sie mit einem Auto-Scaling-Gruppenkapazitätsanbieter fertig sind, können Sie ihn löschen. Nachdem die Gruppe gelöscht wurde, wechselt der Auto Scaling Scaling-Gruppenkapazitätsanbieter in den `INACTIVE` Status. Kapazitätsanbieter mit dem Status `INACTIVE` können für einen bestimmten Zeitraum im Konto erkennbar bleiben. Dieses Verhalten kann sich jedoch in Zukunft ändern und Sie sollten sich nicht darauf verlassen, dass `INACTIVE`-Kapazitätsanbieter bestehen bleiben. Bevor der Kapazitätsanbieter der Auto-Scaling-Gruppe gelöscht wird, muss der Kapazitätsanbieter aus der Kapazitätsanbieter-Strategie aller Services entfernt werden. Sie können die `UpdateService`-API oder den Update-Service-Workflow in der Amazon-ECS-Konsole verwenden, um einen Kapazitätsanbieter aus der Kapazitätsanbieter-Strategie eines Service zu entfernen. Verwenden Sie die Option Neue Bereitstellung erzwingen, um sicherzustellen, dass alle Aufgaben, die die vom Kapazitätsanbieter bereitgestellte Amazon EC2 EC2-Instance-Kapazität nutzen, auf die Kapazität der verbleibenden Kapazitätsanbieter umgestellt werden.

So löschen Sie einen Kapazitätsanbieter für den Cluster (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster: ***Name*** die Option Infrastructure (Infrastruktur), die Auto-Scaling-Gruppe und dann Delete (Löschen) aus.
5. Geben Sie im Bestätigungsfeld delete ***Auto-Scaling-Gruppenname löschen*** ein
6. Wählen Sie Löschen aus.

Abmeldung einer Amazon ECS-Container-Instance

> **Important**
>
> Dieses Thema gilt nur für Container-Instances, die in Amazon EC2 erstellt wurden. Weitere Informationen über das Abmelden externer Instances finden Sie unter Deregistrierung einer externen Amazon ECS-Instance.

Wenn Sie eine von Amazon EC2 gestützte Container-Instance nicht mehr benötigen, können Sie ihre Registrierung im Cluster aufheben. Nach Aufhebung der Registrierung kann die Container-Instance keine neuen Aufgaben mehr akzeptieren.

Wenn zum Zeitpunkt des Aufhebens der Registrierung auf der Container-Instance Aufgaben ausgeführt werden, bleiben diese Aufgaben aktiv, bis Sie die Instance beenden oder die Aufgaben auf andere Weise gestoppt werden. Diese Aufgaben sind jedoch verwaist, d. h. sie werden von Amazon ECS nicht mehr überwacht bzw. erfasst. Falls eine verwaiste Aufgabe auf Ihrer Container-Instance Teil eines Amazon-ECS-Service ist, startet der Service-Scheduler eine andere Kopie der betreffenden Aufgabe in einer anderen Container-Instance, sofern dies möglich ist. Alle Container in verwaisten Dienstaufgaben, die bei einer Application-Load-Balancer-Zielgruppe registriert sind, werden aufgehoben. Sie beginnen mit dem Connection Draining gemäß den Einstellungen auf dem Load Balancer oder der Zielgruppe. Wenn eine verwaiste Aufgabe den `awsvpc`-Netzwerkmodus verwendet, werden ihre Elastic-Network-Schnittstellen gelöscht.

Wenn Sie die Container-Instance nach der Aufhebung der Registrierung zu einem anderen Zweck verwenden möchten, sollten Sie alle auf der Container-Instance ausgeführten Tasks vor Aufhebung

der Registrierung stoppen. Auf diese Weise hören alle verwaisten Aufgaben auf, Ressourcen zu verbrauchen.

Beachten Sie beim Abmelden einer Container-Instance die folgenden Überlegungen.

- Da es zu jeder Container-Instance eindeutige Statusinformationen gibt, sollte die Registrierung von Container-Instances nicht in einem Cluster aufgehoben und in einem anderen Cluster neu vorgenommen werden. Um Ressourcen von Container-Instances zu verschieben, empfehlen wir Ihnen, die Container-Instances aus einem Cluster zu beenden und neue Container-Instances in dem neuen Cluster zu starten. Weitere Informationen finden Sie unter Terminate your Instance im Amazon EC2 EC2-Benutzerhandbuch undStarten einer Amazon ECS Linux-Container-Instance.
- Wenn die Container-Instance von einer Auto Scaling Scaling-Gruppe oder einem AWS CloudFormation Stack verwaltet wird, beenden Sie die Instance, indem Sie die Auto Scaling-Gruppe oder den Auto AWS CloudFormation Scaling-Stack aktualisieren. Andernfalls wird die Auto-Scaling-Gruppe oder AWS CloudFormation eine neue Instance erstellen, nachdem Sie sie beenden.
- Wenn Sie eine aktive Container-Instance mit verbundenem Amazon-ECS-Container-Agent beenden, hebt der Agent die Registrierung der Instance in Ihrem Cluster automatisch auf. Für gestoppte Container-Instances oder Instances mit nicht verbundenen Agents wird die Registrierung bei der Beendigung nicht automatisch aufgehoben.
- Beim Aufheben der Registrierung einer Container-Instance wird die Instance aus dem Cluster entfernt, die Amazon-EC2-Instance wird jedoch nicht beendet. Wenn Sie die Instance nicht mehr benötigen, müssen Sie sie beenden, damit keine weiteren Kosten abgerechnet werden. Weitere Informationen finden Sie unter Beenden Ihrer Instance im Amazon-EC2-Benutzerhandbuch.

Verfahren

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre externe Instance registriert ist.
3. Wählen Sie im Navigationsbereich Clusters und danach den Cluster aus, der Ihre Instance hostet.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur).
5. Wählen Sie unter Container instances (Container-Instances) die Instance-ID aus, deren Anmeldung aufgehoben werden soll. Sie werden zur Detailseite für Container-Instance umgeleitet.

6. Wählen Sie auf der Seite Container Instance : ***id*** Deregister aus.
7. Wählen Sie auf dem Bestätigungsbildschirm Registrierung aufheben.
8. Wenn Sie die Container-Instance nicht mehr benötigen, beenden Sie die zugrunde liegende Amazon-EC2-Instance. Weitere Informationen finden Sie unter Terminate Your Instance im Amazon EC2 EC2-Benutzerhandbuch.

Entleeren von Amazon ECS-Container-Instances

Es kann vorkommen, dass Sie eine Container-Instance aus Ihrem Cluster entfernen müssen, um beispielsweise Systemaktualisierungen durchzuführen oder die Clusterkapazität zu reduzieren. Amazon ECS bietet die Möglichkeit, eine Container-Instance in einen `DRAINING`-Status zu überführen. Dies wird Container-Instance-Ausgleich genannt. Wenn eine Container-Instance auf `DRAINING` festgelegt wird, lässt es Amazon ECS nicht zu, dass die Platzierung neuer Aufgaben in der Container-Instance geplant wird.

Ausgleichsverhalten für Services

Alle Aufgaben, die Teil eines Dienstes sind, die sich in einem `PENDING`-Zustand befinden, werden sofort gestoppt. Wenn im Cluster verfügbare Kapazität für Container-Instances vorhanden ist, startet der Service-Scheduler Ersetzungsaufgaben. Wenn nicht genügend Kapazität für Container-Instances vorhanden ist, wird eine Service-Ereignismeldung gesendet, die das Problem angibt.

Aufgaben, die Teil eines Dienstes auf der Container-Instance sind, die sich in einem `RUNNING`-Zustand befinden werden in einen `STOPPED`-Zustand übertragen. Der Service-Planer versucht, die Aufgaben gemäß dem Bereitstellungstyp und den Konfigurationsparametern `minimumHealthyPercent` und `maximumPercent` des Services zu ersetzen. Weitere Informationen finden Sie unter Amazon-ECS-Dienstleistungen und Parameter der Amazon ECS-Servicedefinition.

- Beträgt der Wert für `minimumHealthyPercent` weniger als 100 % kann der Scheduler die Angabe `desiredCount` während des Ersetzens der Aufgabe vorübergehend ignorieren. Beträgt der Wert für `desiredCount` beispielsweise vier Aufgaben, kann der Scheduler bei einem Minimum von 50 % zwei bestehende Aufgaben stoppen, bevor er zwei neue Aufgaben startet. Bei einem Minimum von 100 % kann der Service-Scheduler keine vorhandenen Aufgaben entfernen, bis die Ersatzaufgaben als fehlerfrei angesehen werden. Wenn Aufgaben für Services, die keinen Load Balancer verwenden, den Status `RUNNING` aufweisen, werden Sie als fehlerfrei angesehen. Aufgaben für Services, die einen Load Balancer nutzen, gelten als fehlerfrei, wenn Sie den Status

RUNNING aufweisen und die Container-Instance, auf der sie gehostet sind, vom Load Balancer als fehlerfrei gemeldet wird.

> **Important**
>
> Wenn Sie Spot-Instances verwenden und `minimumHealthyPercent` größer oder gleich 100 % ist, hat der Service nicht genug Zeit, um die Aufgabe zu ersetzen, bevor die Spot-Instance beendet wird.

- Der Parameter `maximumPercent` stellt eine Obergrenze für die Anzahl der laufenden Aufgaben während der Aufgabenersetzung dar, sodass Sie die Größe des Ersatzstapels festlegen können. Bei einem `desiredCount` von vier Aufgaben beispielsweise werden bei einem Maximum von 200 % vier neue Aufgaben gestartet, bevor die vier auszugleichenden Aufgaben gestoppt werden (sofern die hierfür erforderlichen Cluster-Ressourcen verfügbar sind). Bei einem Maximum von 100 % können keine Ersatzaufgaben gestartet werden, bis die Ausgleichsaufgaben gestoppt wurden.

  > **Important**
  >
  > Wenn sowohl `minimumHealthyPercent` als auch `maximumPercent` 100 % betragen, kann der Service vorhandene Aufgaben nicht entfernen und auch keine Ersatzaufgaben starten. Dies verhindert einen erfolgreichen Ausgleich von Container-Instances und verhindert neue Bereitstellungen.

Ausgleichsverhalten für eigenständige Aufgaben

Alle eigenständigen Aufgaben im `PENDING`- oder `RUNNING`-Status bleiben unberührt. Sie müssen warten, bis sie von alleine stoppen, oder müssen sie manuell stoppen. Die Container-Instance bleibt im Status `DRAINING`.

Eine Container-Instance hat den Ausgleich abgeschlossen, wenn alle Aufgaben, die auf der Instance ausgeführt werden, zu einem `STOPPED`-Zustand übergegangen sind. Die Container-Instance verbleibt in einem `DRAINING`-Zustand, bis sie erneut aktiviert oder gelöscht wird. Sie können den Status der Aufgaben auf der Container-Instance überprüfen, indem Sie den ListTasksVorgang mit dem `containerInstance` Parameter verwenden, um eine Liste von Aufgaben auf der Instance abzurufen, gefolgt von einem DescribeTasksVorgang mit dem Amazon-Ressourcennamen (ARN) oder der ID jeder Aufgabe, um den Aufgabenstatus zu überprüfen.

Wenn Sie bereit sind, dass die Container-Instance erneut mit dem Hosten von Tasks beginnen kann, ändern Sie den Status der Container-Instance von `DRAINING` auf `ACTIVE`. Der Amazon-ECS-Service Scheduler berücksichtigt dann die Container-Instance für die Aufgabenplatzierung erneut.

Verfahren

Die folgenden Schritte können verwendet werden, um eine Container-Instance mit der neuen AWS Management Console zum Ausgleich einzustellen.

Sie können auch die UpdateContainerInstancesStateAPI-Aktion oder den Befehl update-container-instances-state verwenden, um den Status einer Container-Instance auf zu ändern. `DRAINING`

AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Seite Clusters einen Cluster aus, der Ihre Instances hostet.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Infrastructure (Infrastruktur). Aktivieren Sie dann unter Container instances (Container-Instances) das Kontrollkästchen für jede Container-Instance, die Sie ausgleichen möchten.
5. Wählen Sie Aktionen, Entleeren.

Linux-Container-Agent von Amazon ECS

Der Amazon ECS-Agent ist ein Prozess, der auf jeder Container-Instance ausgeführt wird, die in Ihrem Cluster registriert ist. Es erleichtert die Kommunikation zwischen Ihren Container-Instances und Amazon ECS.

Jede Amazon-ECS-Container-Agenten-Version unterstützt unterschiedliche Features und bietet Fehlerbehebungen aus früheren Versionen. Wir empfehlen grundsätzlich, die neueste Version des Amazon-ECS-Container-Agenten zu verwenden, wenn dies möglich ist. Die neueste Version für Ihren Container-Agenten finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Unter https://github.com/aws/amazon-ecs-agent/releases finden Sie alle Features und Verbesserungen jeder Agenten-Version.

> **Important**
>
> Die Mindestversion von Docker für zuverlässige Metriken ist die Docker-Version `v20.10.13` und neuer, die in Amazon-ECS-optimiertem AMI `20220607` und neuer enthalten ist.
> Die Amazon-ECS-Agent-Versionen `1.20.0` und neuer haben die Unterstützung für Docker-Versionen älter als `1.9.0` eingestellt.

Lebenszyklus

Wenn der Amazon-ECS-Container-Agent eine Amazon-EC2-Instance an Ihrem Cluster registriert, meldet die Amazon-EC2-Instance den Status als `ACTIVE` und den Agent-Verbindungsstatus als `TRUE`. Die Container-Instance kann Anfragen zur Ausführung von Aufgaben akzeptieren.

Wenn Sie eine Container-Instance anhalten (nicht beenden), lautet ihr Status weiterhin `ACTIVE`, der Agent-Verbindungsstatus wird jedoch innerhalb von wenigen Minuten in `FALSE` geändert. Alle Aufgaben, die gerade auf der Container-Instance ausgeführt wurden, werden gestoppt. Wenn Sie die Container-Instance erneut starten, stellt der Container-Agent erneut eine Verbindung zum Amazon-ECS-Service her und Sie können wieder Aufgaben auf der Instance ausführen.

> **Important**
>
> Wenn Sie eine Container-Instance stoppen und starten oder einen Neustart der Instance durchführen, wird die Instance bei einigen älteren Versionen des Amazon-ECS-Container-Agenten erneut registriert, ohne dass die Registrierung der ursprünglichen Container-Instance-ID aufgehoben wird. In diesem Fall listet Amazon ECS mehr Container-Instances in Ihrem Cluster auf, als tatsächlich vorhanden sind. (Wenn Sie doppelte Container-Instance-IDs für dieselbe Amazon-EC2-Instance-ID haben, können Sie die Registrierung der als `ACTIVE` aufgeführten Duplikate mit dem Agent-Verbindungsstatus `FALSE` ohne Probleme aufheben.) Dieses Problem ist in der aktuellen Version des Amazon-ECS-Container-Agenten behoben. Weitergehende Informationen zum Aktualisieren auf die aktuelle Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Wenn Sie den Status einer Container-Instance in `DRAINING` ändern, werden keine neuen Aufgaben in die Container-Instance gestellt. Wenn möglich, werden alle auf der Container-Instance ausgeführten Serviceaufgaben entfernt, damit Sie Systemaktualisierungen vornehmen können. Weitere Informationen finden Sie unter Entleeren von Amazon ECS-Container-Instances.

Wenn Sie die Registrierung einer Container-Instance aufheben oder eine Container-Instance beenden, wird ihr Status umgehend in `INACTIVE` geändert und die Container-Instance ist nicht mehr in der Auflistung Ihrer Container-Instances aufgeführt. Allerdings ist nach der Beendigung noch für eine Stunde eine Beschreibung der Container-Instance möglich. Nach einer Stunde ist die Instance-Beschreibung nicht mehr verfügbar.

> **Important**
>
> Sie können die Instances manuell ausgleichen oder einen Auto-Scaling-Gruppenlebenszyklus-Hook erstellen, um den Instance-Status auf `DRAINING` zu setzen. Weitere Informationen zu Auto-Scaling-Lebenszyklus-Hooks finden Sie unter Lebenszyklus-Hooks für Amazon EC2 Auto Scaling.

Amazon-ECS-optimiertes AMI

Die Linux-Varianten des Amazon-ECS-optimierten AMI verwenden das Amazon-Linux-2-AMI als Basis. Das Amazon-Linux-2-Quell-AMI für jede Variante kann durch Abfragen der Parameter-Store-API von Systems Manager abgerufen werden. Weitere Informationen finden Sie unter Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten. Wenn Sie Ihre Container-Instances mit dem neuesten für Amazon ECS optimierten Amazon-Linux-2-AMI starten, erhalten Sie die aktuelle Version des Container-Agenten. Weitere Informationen zum Starten einer Container-Instance mit dem neuesten Amazon-ECS-optimierten Amazon Linux 2-AMI finden Sie unterStarten einer Amazon ECS Linux-Container-Instance.

Zusätzliche Informationen

Auf den folgenden Seiten finden Sie weitere Informationen zu den Änderungen:

- Amazon ECS Agent-Änderungsprotokoll aktiviert GitHub
- Der Quellcode für die `ecs-init`-Anwendung sowie die Skripts und die Konfiguration für das Paketieren des Agenten sind jetzt Teil des Agent-Repositorys. Ältere Versionen von `ecs-init` und deren Verpackung finden Sie im Amazon ecs-init Changelog unter GitHub
- Versionshinweise zu Amazon Linux 2.
- Versionshinweise für die Docker-Engine in der Docker-Dokumentation
- NVIDIA-Treiberdokumentation in der NVIDIA-Dokumentation

Konfiguration des Amazon-ECS-Container-Agenten

Der Amazon ECS-Container-Agent unterstützt eine Reihe von Konfigurationsoptionen, von denen Sie die meisten über Umgebungsvariablen festlegen.

Wenn Ihre Container-Instance mit einer Linux-Variante des Amazon-ECS-optimierten AMI gestartet wurde, können Sie diese Umgebungsvariablen in der Datei `/etc/ecs/ecs.config` setzen und dann den Agenten neu starten. Sie können diese Konfigurationsvariablen auch beim Starten mit Amazon EC2-Benutzerdaten in ihre Container-Instances schreiben. Weitere Informationen finden Sie unter Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe.

Wenn Ihre Container-Instance mit einer Windows-Variante des Amazon ECS-optimierten AMI gestartet wurde, können Sie diese Umgebungsvariablen mit dem PowerShell SetEnvironmentVariable Befehl festlegen und dann den Agenten neu starten. Weitere Informationen finden Sie unter Befehle auf Ihrer Windows-Instance beim Start ausführen im Amazon EC2 EC2-Benutzerhandbuch undthe section called “Bootstrapping von Container-Instances”.

Wenn Sie den Amazon-ECS-Container-Agenten manuell starten (bei AMIs, die nicht Amazon-ECS-optimiert sind), können Sie diese Umgebungsvariablen in dem Befehl docker run verwenden, den Sie benutzen, um den Agent zu starten. Verwenden Sie diese Variablen mit der Syntax `--env=`*`VARIABLE_NAME`*`=`*`VARIABLE_VALUE`*. Bei sensiblen Informationen, beispielsweise Authentifizierungs-Anmeldeinformationen für private Repositorys, sollten Sie die Agent-Umgebungsvariablen in einer Datei speichern und sie alle auf einmal mit der Option `--env-file` *`path_to_env_file`* weitergeben. Sie können die folgenden Befehle verwenden, um die Variablen hinzuzufügen.

```
sudo systemctl stop ecs
sudo vi /etc/ecs/ecs.config
# And add the environment variables with VARIABLE_NAME=VARIABLE_VALUE format.
sudo systemctl start ecs
```

Verfügbare Parameter

Informationen zu den verfügbaren Amazon ECS-Container-Agent-Konfigurationsparametern finden Sie unter Amazon ECS Container Agent on GitHub.

Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3

Die Konfiguration des Amazon ECS-Container-Agenten wird mit der Umgebungsvariablen gesteuert. Die Linux-Varianten des Amazon-ECS-optimierten AMI suchen nach diesen Variablen in `/etc/`

`ecs/ecs.config`, wenn der Containeragent startet, und konfigurieren den Agenten entsprechend. Bestimmte unverfängliche Umgebungsvariablen, beispielsweise `ECS_CLUSTER`, können beim Starten durch Amazon EC2-Benutzerdaten an die Container-Instance übergeben und folgenlos in diese Datei geschrieben werden. Andere vertrauliche Informationen, wie Ihre AWS Anmeldeinformationen oder die `ECS_ENGINE_AUTH_DATA` Variable, sollten jedoch niemals in Form von Benutzerdaten an eine Instance weitergegeben oder so geschrieben werden, dass sie in einer `.bash_history` Datei angezeigt werden können. `/etc/ecs/ecs.config`

Das Speichern von Konfigurationsinformationen in einem privaten Amazon S3-Bucket und das Erteilen der schreibgeschützten Zugriffsberechtigung Ihrer IAM-Rolle der Container-Instance ist eine sichere und bequeme Art, die Konfiguration der Container-Instance zur Startzeit zu ermöglichen. Sie können eine Kopie Ihrer `ecs.config`-Datei in einem privaten Bucket speichern. Sie können dann Amazon EC2 EC2-Benutzerdaten verwenden, um die Instance zu installieren AWS CLI und Ihre Konfigurationsinformationen zu kopieren, `/etc/ecs/ecs.config` wenn die Instance gestartet wird.

So speichern Sie eine **`ecs.config`**-Datei in Amazon S3

1. Sie müssen der Container-Instance-Rolle (ecs InstanceRole) Berechtigungen erteilen, um nur Lesezugriff auf Amazon S3 zu haben. Sie können dies tun, indem Sie der Rolle den ReadOnlyAmazonS3-Zugriff zuweisen. `ecsInstanceRole` Informationen zum Anhängen einer Richtlinie an eine Rolle finden Sie unter Ändern einer Rollenberechtigungsrichtlinie (Konsole) im Benutzerhandbuch AWS Identity and Access Management
2. Erstellen Sie eine `ecs.config`-Datei mit gültigen Amazon-ECS-Agenten-Konfigurationsvariablen im folgenden Format. Dieses Beispiel konfiguriert private Registry-Authentifizierung. Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.

   ```
   ECS_ENGINE_AUTH_TYPE=dockercfg
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
   ```

   > **Note**
   >
   > Eine vollständige Liste der verfügbaren Amazon ECS-Agenten-Konfigurationsvariablen finden Sie unter Amazon ECS Container Agent on GitHub.

3. Um Ihre Konfigurationsdatei zu speichern, erstellen Sie in Amazon S3 einen privaten Bucket. Weitere Informationen finden Sie unter Erstellen eines Buckets im Benutzerhandbuch zu Amazon Simple Storage Service.
4. Laden Sie die Datei `ecs.config` in Ihren S3-Bucket hoch. Anleitungen finden Sie unter Hinzufügen eines Objekts zu einem Bucket im Benutzerhandbuch zu Amazon Simple Storage Service.

So laden Sie eine **`ecs.config`**-Datei beim Start von Amazon S3

1. Führen Sie in diesem Bereich die zuvor genannten Verfahren aus, um schreibgeschützten Amazon S3-Zugriff auf Ihre Container-Instances zu ermöglichen und eine `ecs.config`-Datei in einem privaten S3-Bucket zu speichern.
2. Starten Sie neue Container-Instances und verwenden Sie das folgende Beispielskript in den EC2-Benutzerdaten. Das Skript installiert die AWS CLI und kopiert Ihre Konfigurationsdatei nach `/etc/ecs/ecs.config`. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Installieren des Amazon-ECS-Container-Agenten

Wenn Sie eine Amazon EC2-Instance bei Ihrem Amazon ECS-Cluster registrieren möchten und diese Instance kein AMI verwendet, das auf dem Amazon ECS-optimierten AMI basiert, können Sie den Amazon ECS-Container-Agenten mithilfe des folgenden Verfahrens manuell installieren. Dazu können Sie den Agenten entweder von einem der regionalen Amazon S3-Buckets oder von Amazon Elastic Container Registry Public herunterladen. Wenn Sie aus einem der regionalen Amazon S3 S3-Buckets herunterladen, können Sie optional die Gültigkeit der Container-Agent-Datei anhand der PGP-Signatur überprüfen.

> **Note**
>
> Die `systemd`-Einheiten für die Amazon ECS- und Docker-Services enthalten die Anweisung, auf die Beendigung von `cloud-init` zu warten, bevor beide Services gestartet werden. Der `cloud-init`-Prozess gilt erst als abgeschlossen, wenn die Ausführung der Amazon EC2-

Benutzerdaten beendet wurde. Daher kann das Starten von Amazon-ECS oder Docker über Amazon-EC2-Benutzerdaten zu einer Systemblockade führen. Um den Container-Agent mit Amazon EC2-Benutzerdaten zu starten, können Sie `systemctl enable --now --no-block ecs.service` verwenden.

So installieren Sie den Amazon-ECS-Container-Agent auf einer Nicht–Amazon Linux-EC2-Instance

Um den Amazon ECS-Container-Agenten auf einer Amazon EC2 EC2-Instance zu installieren, können Sie den Agenten aus einem der regionalen Amazon S3 S3-Buckets herunterladen und installieren.

Note

Wenn Sie ein Nicht-Amazon-Linux-AMI verwenden, erfordert Ihre Amazon-EC2-Instance `cgroupfs`-Support für die `cgroup`-Treiber, damit der Amazon-ECS-Agent Ressourcenlimits auf Aufgabenebene unterstützt. Weitere Informationen finden Sie unter Amazon ECS-Agent unter GitHub.

Die neuesten Dateien des Container-Agents von Amazon ECS nach Region für jede Systemarchitektur sind unten als Referenz aufgeführt.

Region	Name der Region	Amazon-ECS-Init-Deb-Dateien	Amazon-ECS-Init-rpm-Dateien
us-east-2	USA Ost (Ohio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-east-1	USA Ost (Nord-Virginia)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Region	Name der Region	Amazon-ECS-Init-Deb-Dateien	Amazon-ECS-Init-rpm-Dateien
us-west-1	USA West (Nordkalifornien)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-2	USA West (Oregon)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-east-1	Asien-Pazifik (Hongkong)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-1	Asien-Pazifik (Tokio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-2	Asien-Pazifik (Seoul)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-south-1	Asien-Pazifik (Mumbai)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Region	Name der Region	Amazon-ECS-Init-Deb-Dateien	Amazon-ECS-Init-rpm-Dateien
ap-southeast-1	Asien-Pazifik (Singapur)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-2	Asien-Pazifik (Sydney)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ca-central-1	Kanada (Zentral)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-central-1	Europa (Frankfurt)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-1	Europa (Irland)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-2	Europa (London)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Region	Name der Region	Amazon-ECS-Init-Deb-Dateien	Amazon-ECS-Init-rpm-Dateien
eu-west-3	Europa (Paris)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
sa-east-1	Südamerika (São Paulo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 Amazon ECS init aarch64 (aarch64)
us-gov-east-1	AWS GovCloud (USA-Ost)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-gov-west-1	AWS GovCloud (US-West)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

So installieren Sie den Amazon-ECS-Container-Agent auf einer Amazon-EC2-Instance mit einem Nicht-Amazon Linux-AMI

1. Starten Sie eine Amazon-EC2-Instance mit einer IAM-Rolle, die den Zugriff auf Amazon ECS erlaubt. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.
2. Verbinden Sie sich mit der Instance.
3. Installieren Sie die neueste Docker-Version auf Ihrer Instance.
4. Prüfen Sie Ihre Docker-Version, um sicherzustellen, dass Ihr System den Anforderungen der Mindestversion entspricht.

> **Note**
>
> Die Mindestversion von Docker für zuverlässige Metriken ist die Docker-Version `v20.10.13` und neuer, die in Amazon-ECS-optimiertem AMI `20220607` und neuer enthalten ist.
> Die Amazon-ECS-Agent-Versionen `1.20.0` und neuer haben die Unterstützung für Docker-Versionen älter als `1.9.0` eingestellt.

```
docker --version
```

5. Laden Sie die entsprechende Amazon-ECS-Agent-Datei für Ihr Betriebssystem und Ihre Systemarchitektur herunter und installieren Sie sie.

   Für `deb`-Architekturen:

```
ubuntu:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/amazon-ecs-init-latest.amd64.deb
ubuntu:~$ sudo dpkg -i amazon-ecs-init-latest.amd64.deb
```

   Für `rpm`-Architekturen:

```
fedora:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/amazon-ecs-init-latest.x86_64.rpm
fedora:~$ sudo yum localinstall -y amazon-ecs-init-latest.x86_64.rpm
```

6. Bearbeiten Sie die `/lib/systemd/system/ecs.service` Datei und fügen Sie am Ende des `[Unit]` Abschnitts die folgende Zeile hinzu.

```
After=cloud-final.service
```

7. (Optional) Um die Instance bei einem anderen Cluster als dem `default`-Cluster anzumelden, bearbeiten Sie die `/etc/ecs/ecs.config`-Datei und fügen Sie den folgenden Inhalt hinzu. Das folgende Beispiel gibt den `MyCluster`-Cluster an.

```
ECS_CLUSTER=MyCluster
```

Weitere Informationen zu diesen und anderen Agenten-Laufzeitoptionen erhalten Sie unterKonfiguration des Amazon-ECS-Container-Agenten.

> **Note**
>
> Sie können Ihre Agenten-Umgebungsvariablen optional in Amazon S3 speichern (diese können in Ihren Container-Instances zum Startzeitpunkt mithilfe von Amazon EC2-Benutzerdaten heruntergeladen werden). Dies empfiehlt sich für sensible Daten, wie beispielsweise Authentifizierungs-Anmeldeinformationen für private Repositorys. Weitere Informationen finden Sie unter Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3 und Verwenden von AWS Nicht-Container-Images in Amazon ECS.

8. Starten Sie den Service `ecs`.

```
ubuntu:~$ sudo systemctl start ecs
```

Ausführen des Amazon-ECS-Agenten mit dem Host-Netzwerkmodus

Beim Ausführen des Amazon-ECS-Container-Agenten erstellt `ecs-init` den Container-Agent-Container mit dem Netzwerkmodus `host`. Dies ist der einzige unterstützte Netzwerkmodus für den Container-Agent-Container.

Dies ermöglicht Ihnen, den Zugriff auf den Amazon-EC2-Instance-Metadaten-Service-Endpunkt (`http://169.254.169.254`) für die vom Container-Agenten gestarteten Container zu blockieren. Auf diese Weise wird sichergestellt, dass Container vom Container-Instance-Profil aus keinen Zugriff auf Anmeldeinformationen der IAM-Rolle haben, und erzwungen, dass die Aufgabe nur die Anmeldeinformationen der IAM-Aufgabenrolle verwendet. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

Dies sorgt auch dafür, dass der Container-Agent nicht um Verbindungen und Netzwerkdatenverkehr auf der `docker0`-Brücke konkurrieren muss.

Konfigurationsparameter für das Amazon ECS-Container-Agent-Protokoll

Der Amazon-ECS-Container-Agent speichert Protokolle auf Ihren Container-Instances.

Bei Container-Agent-Version 1.36.0 und höher befinden sich die Protokolle standardmäßig unter `/var/log/ecs/ecs-agent.log` auf Linux-Instances und unter `C:\ProgramData\Amazon\ECS\log\ecs-agent.log` auf Windows-Instances.

Bei Container-Agent-Version 1.35.0 und früher befinden sich die Protokolle standardmäßig unter `/var/log/ecs/ecs-agent.log.`*`timestamp`* auf Linux-Instances und unter `C:\ProgramData\Amazon\ECS\log\ecs-agent.log.`*`timestamp`* auf Windows-Instances.

Standardmäßig werden die Agent-Protokolle stündlich rotiert, wobei maximal 24 Protokolle gespeichert werden.

Im Folgenden finden Sie die Konfigurationsvariablen des Container-Agenten, die verwendet werden können, um das standardmäßige Agenten-Protokollierungsverhalten zu ändern. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

`ECS_LOGFILE`

Beispielwerte: `/ecs-agent.log`

Standardwert unter Linux: Null

Standardwert unter Windows: Null

Bestimmt den Speicherort, an dem Agenten-Protokolle geschrieben werden sollen. Wenn Sie den Agenten über ausführen`ecs-init`, was bei der Verwendung des Amazon ECS-optimierten AMI die Standardmethode ist`/log`, lautet der Pfad innerhalb des Containers und hängt diesen auf `/var/log/ecs/` dem `ecs-init` Host ein.

`ECS_LOGLEVEL`

Beispielwerte: `crit`, `error`, `warn`, `info`, `debug`

Standardwert unter Linux: `info`

Standardwert unter Windows: `info`

Die zu protokollierende Detailebene.

`ECS_LOGLEVEL_ON_INSTANCE`

Beispielwerte: `none`, `crit`, `error`, `warn`, `info`, `debug`

Standardwert unter Linux: `none`, wenn `ECS_LOG_DRIVER` explizit auf einen nicht leeren Wert gesetzt wird, andernfalls der gleiche Wert wie `ECS_LOGLEVEL`

Standardwert unter Windows: `none`, wenn `ECS_LOG_DRIVER` explizit auf einen nicht leeren Wert gesetzt wird, andernfalls der gleiche Wert wie `ECS_LOGLEVEL`

Kann verwendet werden, um `ECS_LOGLEVEL` zu überschreiben und um eine Detailebene festzulegen, die in der On-Instance-Protokolldatei protokolliert werden soll, unabhängig von der Ebene, die im Protokollierungstreiber protokolliert wird. Wenn ein Protokollierungstreiber explizit festgelegt ist, sind die On-Instance-Logs standardmäßig deaktiviert. Sie können mit dieser Variablen wieder aktiviert werden.

`ECS_LOG_DRIVER`

Beispielwerte: `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries` `syslog`, `splunk`

Standardwert unter Linux: `json-file`

Der Standardwert unter Windows: Nicht zutreffend

Ermittelt den Protokollierungstreiber, den der Agent-Container verwendet.

`ECS_LOG_ROLLOVER_TYPE`

Beispielwerte: `size`, `hourly`

Standardwert unter Linux: `hourly`

Standardwert unter Windows: `hourly`

Legt fest, ob die Protokolldatei des Container-Agenten stündlich oder nach Größe rotiert wird. Standardmäßig wird die Agent-Protokolldatei stündlich rotiert.

`ECS_LOG_OUTPUT_FORMAT`

Beispielwerte: `logfmt`, `json`

Standardwert unter Linux: `logfmt`

Standardwert unter Windows: `logfmt`

Bestimmt das Protokollausgabeformat. Wenn das `json` Format verwendet wird, ist jede Zeile im Protokoll eine strukturierte JSON-Map.

`ECS_LOG_MAX_FILE_SIZE_MB`

Beispielwerte: `10`

Standardwert unter Linux: `10`

Standardwert unter Windows: `10`

Wenn die `ECS_LOG_ROLLOVER_TYPE` Variable auf gesetzt ist`size`, bestimmt diese Variable die maximale Größe (in MB) der Protokolldatei, bevor sie rotiert wird. Wenn der Rollover-Typ auf `hourly` festgelegt ist, wird diese Variable ignoriert.

`ECS_LOG_MAX_ROLL_COUNT`

Beispielwerte: `24`

Standardwert unter Linux: `24`

Standardwert unter Windows: `24`

Bestimmt die Anzahl der rotierten Protokolldateien, die beibehalten werden sollen. Ältere Protokolldateien werden gelöscht, nachdem diese Grenze erreicht ist.

Für Container-Agent Version 1.36.0 und höher folgt eine Beispielprotokolldatei für die Verwendung des `logfmt`-Formats.

```
level=info time=2019-12-12T23:43:29Z msg="Loading configuration" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-agent:latest" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-pause:0.1.0" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Amazon ECS agent Version: 1.36.0, Commit:
 ca640387" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Creating root ecs cgroup: /ecs"
 module=init_linux.go
level=info time=2019-12-12T23:43:29Z msg="Creating cgroup /ecs"
 module=cgroup_controller_linux.go
level=info time=2019-12-12T23:43:29Z msg="Loading state!" module=statemanager.go
level=info time=2019-12-12T23:43:29Z msg="Event stream ContainerChange start
 listening..." module=eventstream.go
level=info time=2019-12-12T23:43:29Z msg="Restored cluster 'auto-robc'" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Restored from checkpoint file. I
 am running as 'arn:aws:ecs:us-west-2:0123456789:container-instance/auto-
robc/3330a8a91d15464ea30662d5840164cd' in cluster 'auto-robc'" module=agent.go
```

Im Folgenden finden Sie eine Beispielprotokolldatei für die Verwendung des JSON-Formats.

```
{"time": "2019-11-07T22:52:02Z", "level": "info", "msg": "Starting Amazon Elastic
 Container Service Agent", "module": "engine.go"}
```

Für Container-Agent-Versionen 1.35.0 und früher gilt das folgende Format der Protokolldatei.

```
2016-08-15T15:54:41Z [INFO] Starting Agent: Amazon ECS Agent - v1.12.0 (895f3c1)
2016-08-15T15:54:41Z [INFO] Loading configuration
2016-08-15T15:54:41Z [WARN] Invalid value for task cleanup duration, will be overridden
 to 3h0m0s, parsed value 0, minimum threshold 1m0s
2016-08-15T15:54:41Z [INFO] Checkpointing is enabled. Attempting to load state
2016-08-15T15:54:41Z [INFO] Loading state! module="statemanager"
2016-08-15T15:54:41Z [INFO] Detected Docker versions [1.17 1.18 1.19 1.20 1.21 1.22]
2016-08-15T15:54:41Z [INFO] Registering Instance with ECS
2016-08-15T15:54:41Z [INFO] Registered! module="api client"
```

Konfiguration von Amazon ECS-Container-Instances für private Docker-Images

Der Amazon-ECS-Container-Agent kann sich unter Verwendung der Standardauthentifizierung bei privaten Registrys authentifizieren. Wenn Sie die Authentifizierung bei privaten Registrierungen aktivieren, können Sie private Docker-Images in Ihren Aufgabendefinitionen verwenden. Dieses Feature wird nur von Aufgaben unterstützt, die den Starttyp EC2 verwenden.

Eine andere Methode zur Aktivierung der Authentifizierung in privaten Registern besteht AWS Secrets Manager darin, Ihre Anmeldedaten für die private Registrierung sicher zu speichern und sie dann in Ihrer Container-Definition zu referenzieren. Auf diese Weise können Ihre Aufgaben Images aus privaten Repositorys verwenden. Diese Methode unterstützt Aufgaben, die den Starttyp EC2 oder Fargate verwenden. Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.

Der Amazon-ECS-Container-Agent sucht nach zwei Umgebungsvariablen, wenn er gestartet wird:

- `ECS_ENGINE_AUTH_TYPE` gibt die Art der gesendeten Authentifizierungsdaten an.
- `ECS_ENGINE_AUTH_DATA` enthält die tatsächlichen Anmeldeinformationen für die Authentifizierung.

Das Amazon-ECS-optimierte AMI durchsucht die `/etc/ecs/ecs.config`-Datei nach diesen Variablen, wenn die Container-Instance gestartet wird, sowie jedes Mal, wenn der Service gestartet

wird (mit dem Befehl sudo start ecs). AMIs, die nicht für Amazon ECS optimiert sind, sollten diese Umgebungsvariablen in einer Datei speichern und mit der `--env-file` *`path_to_env_file`*-Option an den docker run-Befehl, der den Container-Agenten startet, weitergeben.

> **Important**
>
> Wir raten davon ab, diese Umgebungsvariablen für die Authentifizierung beim Start der Instance mit Amazon EC2-Benutzerdaten einzufügen oder sie mit der Option `--env` an den Befehl docker run zu übergeben. Diese Methoden eigenen sich nicht für vertrauliche Daten wie z. B. Anmeldeinformationen für die Authentifizierung. Anweisungen dazu, wie Sie die Anmeldeinformationen für die Authentifizierung Ihrer Container-Instance sicher hinzufügen, finden Sie unter Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3.

Authentifizierungsformate

Es gibt zwei Formate der Authentifizierung bei privaten Registrierungen: `dockercfg` und `docker`.

dockercfg-Authentifizierungsformat

Das Format `dockercfg` verwendet die in der Konfigurationsdatei gespeicherten Authentifizierungsinformationen. Diese Konfigurationsdatei wird erstellt, wenn Sie den Befehl docker login ausführen. Sie können diese Datei durch die Ausführung von docker login auf Ihrem lokalen System erstellen, wobei Sie den registrierten Benutzernamen, das Passwort und die E-Mail-Adresse eingeben. Sie können sich auch bei einer Container-Instance anmelden und den Befehl dort ausführen. Je nachdem, welche Docker-Version Sie verwenden, wird diese Datei entweder als `~/.dockercfg` oder `~/.docker/config.json` gespeichert.

```
cat ~/.docker/config.json
```

Ausgabe:

```
{
  "auths": {
    "https://index.docker.io/v1/": {
      "auth": "zq212MzEXAMPLE7o6T25Dk0i"
    }
  }
```

```
}
```

> **Important**
>
> Bei neueren Docker-Versionen wird, wie oben zu sehen, eine Konfigurationsdatei mit einem äußeren `auths`-Objekt erstellt. Der Amazon-ECS-Agent unterstützt nur `dockercfg`-Authentifizierungsdaten im folgenden Format, ohne das `auths`-Objekt. Wenn das Dienstprogramm jq installiert ist, können Sie diese Daten mit dem folgenden Befehl extrahieren: cat ~/.docker/config.json | jq .auths.

```
cat ~/.docker/config.json | jq .auths
```

Ausgabe:

```
{
  "https://index.docker.io/v1/": {
    "auth": "zq212MzEXAMPLE7o6T25Dk0i",
    "email": "email@example.com"
  }
}
```

Im obigen Beispiel sollte die folgende Umgebungsvariable der Umgebungsvariablendatei (`/etc/ecs/ecs.config` für das Amazon-ECS-optimierte AMI), die der Amazon-ECS-Container-Agent während der Laufzeit lädt, hinzugefügt werden. Wenn Sie das Amazon-ECS-optimierte AMI nicht verwenden und den Agenten manuell mit dem Befehl docker run starten, müssen Sie die Umgebungsvariablendatei beim Starten des Agenten mit der Option `--env-file` *`path_to_env_file`* festlegen.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

Sie können mehrere private Registrierungen mit der folgenden Syntax konfigurieren:

```
ECS_ENGINE_AUTH_TYPE=dockercfg
```

```
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example-01.com"},"repo.example-02.com":
{"auth":"fQ172MzEXAMPLEoF7225DU0j","email":"email@example-02.com"}}
```

docker-Authentifizierungsformat

Das `docker`-Format verwendet eine JSON-Darstellung des Registrierungsservers, auf dem sich der Agent authentifizieren soll. Außerdem enthält es die Authentifizierungsparameter, die für diese Registrierung benötigt werden (z. B. Benutzername, Passwort und E-Mail-Adresse für dieses Konto). Für ein Docker Hub-Konto sieht die JSON-Darstellung wie folgt aus:

```
{
  "https://index.docker.io/v1/": {
    "username": "my_name",
    "password": "my_password",
    "email": "email@example.com"
  }
}
```

In diesem Beispiel sollte die folgende Umgebungsvariable der Umgebungsvariablendatei (`/etc/ecs/ecs.config` für das Amazon-ECS-optimierte AMI), die der Amazon-ECS-Container-Agent während der Laufzeit lädt, hinzugefügt werden. Wenn Sie das Amazon-ECS-optimierte AMI nicht verwenden und den Agenten manuell mit dem Befehl docker run starten, müssen Sie die Umgebungsvariablendatei beim Starten des Agenten mit der Option `--env-file` *`path_to_env_file`* festlegen.

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
```

Sie können mehrere private Registrierungen mit der folgenden Syntax konfigurieren:

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"username":"my_name","password":"my_password","email":"email@example-01.com"},"repo.example-02
{"username":"another_name","password":"another_password","email":"email@example-02.com"}}
```

Verfahren

Gehen Sie wie folgt vor, um private Registrierungen für Ihre Container-Instances zu aktivieren.

So aktivieren Sie private Registrierungen im Amazon-ECS-optimierten AMI

1. Melden Sie sich mit SSH an Ihrer Container-Instance an.
2. Öffnen Sie die Datei `/etc/ecs/ecs.config` und fügen Sie die Werte `ECS_ENGINE_AUTH_TYPE` und `ECS_ENGINE_AUTH_DATA` für Ihre Registrierung und Ihr Konto hinzu:

   ```
   sudo vi /etc/ecs/ecs.config
   ```

   Dieses Beispiel authentifiziert ein Docker Hub-Benutzerkonto:

   ```
   ECS_ENGINE_AUTH_TYPE=docker
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"username":"my_name","password":"my_password","email":"email@example.com"}}
   ```
3. Überprüfen Sie, ob ihr Agent die Umgebungsvariable `ECS_DATADIR` nutzt, um seinen Status zu speichern:

   ```
   docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Ausgabe:

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Wenn der vorherige Befehl die Umgebungsvariable `ECS_DATADIR` nicht zurücksendet, müssen Sie sämtliche Aufgaben, die auf dieser Container-Instance ausgeführt werden, abbrechen, bevor Sie Ihren Agenten stoppen. Neuere Agenten mit der Umgebungsvariable `ECS_DATADIR` speichern ihren Status und Sie können sie stoppen und starten, während Aufgaben problemlos ausgeführt werden. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.
4. Stoppen Sie den Service `ecs`:

   ```
   sudo stop ecs
   ```

   Ausgabe:

```
ecs stop/waiting
```

5. Den Service `ecs` neu starten.

   - Für das Amazon-ECS-optimierte Amazon Linux 2-AMI:

     ```
     sudo systemctl restart ecs
     ```

   - Für das Amazon-ECS-optimierte Amazon Linux AMI:

     ```
     sudo stop ecs && sudo start ecs
     ```

6. (Optional) Durch Abfragen der Agenten-Introspektions-API-Operation können Sie überprüfen, ob der Agent ausgeführt wird und Sie können Informationen über Ihre neue Container-Instance einholen. Weitere Informationen finden Sie unter the section called “Introspektion von Containern”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Automatische Amazon ECS-Aufgabe und -Image-Bereinigung

Jedes Mal, wenn eine Aufgabe auf einer Container-Instance platziert wird, prüft der Amazon-ECS-Container-Agent, ob die Images, auf die in der Aufgabe verwiesen wird, die neuesten Images des im Repository spezifizierten Tag sind. Wenn dies nicht der Fall ist, gestattet das Standardverhalten dem Agenten, die Images aus den jeweiligen Repositorys abrufen. Wenn Sie die Images in Aufgaben und Services häufig aktualisieren, kann sich der Container-Instance-Speicher schnell mit Docker-Images füllen, die Sie nicht mehr benötigen. Beispielsweise verwenden Sie vielleicht eine Pipeline für fortlaufende Integration und fortlaufende Bereitstellung (Continous Integration and continous deployment (CI/CD)).

> **Note**
>
> Das Verhalten des Amazon-ECS-Agenten beim Abrufen von Images kann mit dem Parameter `ECS_IMAGE_PULL_BEHAVIOR` angepasst werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Ebenso können Container, die zu gestoppten Aufgaben gehören, mit ihren Protokollinformationen, Datenvolumen und anderen Artefakten Container-Instance-Speicherplatz einnehmen. Diese Artefakte eignen sich für das Debugging von Containern, die unerwarteterweise gestoppt wurden. Doch der Großteil dieses Speichers kann nach einem gewissen Zeitraum problemlos freigemacht werden.

Standardmäßig bereinigt der Amazon-ECS-Container-Agent automatisch gestoppte Aufgaben und Docker-Images, die nicht von Aufgaben auf Ihren Container-Instances verwendet werden.

> **Note**
>
> Das Feature der automatischen Image-Bereinigung erfordert mindestens die Amazon-ECS-Container-Agenten-Version 1.13.0. Die neueste Version für Ihren Agent finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Mit den folgenden Agentenkonfigurationsvariablen können Sie Ihre automatische Aufgaben- und Image-Bereinigung an Ihre Bedürfnisse anpassen. Weitere Informationen zum Einrichten dieser Variablen auf Ihren Container-Instances finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

`ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`

Diese Variable gibt die Zeit an, die gewartet wird, bevor Container, die zu gestoppten Aufgaben gehören, entfernt werden. Der Image-Bereinigungsvorgang kann kein Image löschen, wenn noch ein Container darauf verweist. Wenn ein Image keine Referenzen zu Containern (entweder gestoppt oder noch in Ausführung) mehr hat, wird dieses Image ein Kandidat für die Bereinigung. Standardmäßig ist dieser Parameter auf 3 Stunden eingestellt. Sie können diesen Zeitraum jedoch auf bis zu 1 Sekunde verkürzen, wenn dies für Ihre Anwendung erforderlich ist. Der Parameter wird ignoriert, wenn Sie den Wert auf weniger als 1 Sekunde festlegen.

`ECS_DISABLE_IMAGE_CLEANUP`

Wenn Sie diese Variable auf `true` einstellen, ist die automatische Image-Bereinigung auf Ihrer Container-Instance deaktiviert und es werden keine Images automatisch entfernt.

`ECS_IMAGE_CLEANUP_INTERVAL`

Diese Variable gibt an, wie häufig der automatische Image-Bereinigungsvorgang prüft, ob zu löschende Images vorhanden sind. Die Standardeinstellung ist 30 Minuten. Sie können diesen Zeitraum jedoch auch bis zu 10 Minuten verkürzen, wenn die Images häufiger gelöscht werden sollen.

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`

Über diese Variable wird festgelegt, wie viel Zeit mindestens zwischen dem Abrufen eines Image und dem Löschen vergehen muss. So wird verhindert, dass gerade erst abgerufene Images gelöscht werden. Die Standardeinstellung ist 1 Stunde.

`ECS_NUM_IMAGES_DELETE_PER_CYCLE`

Diese Variable gibt an, wie viele Images bei einem einzigen Bereinigungszyklus entfernt werden. Der Standardwert ist 5 und der Minimumwert beträgt 1.

Wenn der Amazon-ECS-Container-Agent ausgeführt wird und die automatische Image-Bereinigung nicht deaktiviert ist, prüft der Agent mit einer Häufigkeit, die durch die Variable `ECS_IMAGE_CLEANUP_INTERVAL` angegeben wird, ob Docker-Images vorhanden sind, auf die nicht von laufenden oder gestoppten Containern verwiesen wird. Werden ungenutzte Images gefunden und diese sind älter als die durch die Variable `ECS_IMAGE_MINIMUM_CLEANUP_AGE` angegebene Mindestbereinigungszeit, entfernt der Agent die maximale Anzahl an Images, die durch die Variable `ECS_NUM_IMAGES_DELETE_PER_CYCLE` festgelegt ist. Das Image, dessen Referenz am längsten her ist, wird als erstes gelöscht. Nach dem Entfernen der Images wartet der Agent das nächste Intervall ab und wiederholt dann den Vorgang.

Planen Sie Ihre Container auf Amazon ECS

Amazon Elastic Container Service (Amazon ECS) ist ein optimistisches Parallelitätssystem mit gemeinsamem Status, das flexible Planungsfunktionen für Ihre containerisierten Workloads bietet. Die Amazon-ECS-Scheduler nutzen dieselben Clusterzustandsinformationen, die von der Amazon-ECS-API bereitgestellt werden, um entsprechende Entscheidungen zur Platzierung zu treffen.

Amazon ECS bietet einen Service-Scheduler für lang laufende Aufgaben und Anwendungen. Es bietet auch die Möglichkeit, eigenständige Aufgaben oder geplante Aufgaben für Batch-Jobs oder einzelne Run-Tasks auszuführen. Sie können die Strategien und Einschränkungen für die Aufgabenplatzierung für ausgeführte Aufgaben angeben, die Ihren Anforderungen am besten entsprechen. Sie können z. B. angeben, ob Aufgaben über mehrere Availability Zones oder innerhalb einer einzelnen Availability Zone ausgeführt werden. Integrieren Sie Aufgaben optional mit Ihren eigenen benutzerdefinierten oder Drittanbieter-Schedulern.

Option	Wann sollte dies verwendet werden?	Weitere Informationen
Service	Der Service Scheduler eignet sich für zustandslose Dienste und Anwendungen mit langer Laufzeit. Optional stellt der Service-Scheduler auch sicher, dass Aufgaben für einen Elastic Load Balancing -Load Balancer registriert werden. Sie können Ihre Services, die vom Service-S cheduler verwaltet werden, aktualisieren. Dies kann das Bereitstellen einer neuen Aufgabendefinition oder das Ändern der Anzahl der gewünschten Aufgaben umfassen, die ausgeführ t werden. Standardmäßig	Amazon-ECS-Dienstl eistungen

Option	Wann sollte dies verwendet werden?	Weitere Informationen
	verteilt der Service Scheduler Aufgaben über mehrere Availability Zones. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie jedoch festlegen, wie Aufgaben platziert und beendet werden.	
Eigenständige Aufgabe	Eine eigenständige Aufgabe eignet sich für Prozesse wie Batch-Jobs, die Arbeit ausführen und dann beenden. Zum Beispiel können Sie einen Prozess `RunTask` aufrufen lassen, wenn ein Auftrag in eine Warteschlange gestellt wird. Die Aufgabe nimmt den Auftrag aus der Warteschlange, führt ihn aus und wird dann beendet. Mithilfe von `RunTask` können Sie der Standardstrategie zur Platzierung von Aufgaben erlauben, Aufgaben zufällig über Ihr Cluster zu verteilen. Dadurch wird die Wahrscheinlichkeit gesenkt, dass einer einzelnen Instance eine unverhältnismäßige Anzahl von Aufgaben zugewiesen wird.	Eigenständige Amazon ECS-Aufgaben

Option	Wann sollte dies verwendet werden?	Weitere Informationen
Geplante Aufgaben	Eine geplante Aufgabe eignet sich, wenn Sie Aufgaben in festgelegten Intervallen in Ihrem Cluster ausführen müssen. Sie können den EventBridge Scheduler verwenden, um einen Zeitplan zu erstellen. Sie können Aufgaben für einen Backup-Vorgang oder einen Protokoll-Scan ausführen. Der von Ihnen EventBridge erstellte Scheduler-Zeitplan kann eine oder mehrere Aufgaben in Ihrem Cluster zu bestimmte n Zeiten ausführen. Ihr geplantes Ereignis kann auf ein bestimmtes Intervall eingestellt werden (alle *N* Minuten, Stunden oder Tage ausführen). Andernfalls können Sie für eine komplizie rtere Planung einen `cron`-Ausdruck verwenden.	Verwenden von Amazon EventBridge Scheduler zur Planung von Amazon ECS-Aufgaben

Berechnungsoptionen

Mit Amazon ECS können Sie die Infrastruktur angeben, auf der Ihre Aufgaben oder Services ausgeführt werden. Sie können eine Kapazitätsanbieterstrategie oder einen Starttyp verwenden.

Für Fargate sind die Kapazitätsanbieter Fargate und Fargate Spot. Für EC2 ist der Kapazitätsanbieter die Auto Scaling Scaling-Gruppe mit den registrierten Container-Instances.

Die Kapazitätsanbieter-Strategie verteilt Ihre Aufgaben auf die Kapazitätsanbieter, die Ihrem Cluster zugeordnet sind.

Nur Kapazitätsanbieter, die bereits einem Cluster zugeordnet sind und den `ACTIVE`- oder `UPDATING`-Status haben, können in einer Kapazitätsanbieter-Strategie verwendet werden. Sie können einen Kapazitätsanbieter einem Cluster zuordnen, wenn Sie einen Cluster erstellen.

In einer Kapazitätsanbieter-Strategie gibt der optionale Basis-Wert an, wie viele Aufgaben mindestens auf einem bestimmten Kapazitätsanbieter ausgeführt werden. In einer Kapazitätsanbieterstrategie kann nur für einen Kapazitätsanbieter ein Basiswert festgelegt werden.

Der Gewichtungs-Wert bestimmt den relativen Prozentsatz der Gesamtzahl gestarteter Aufgaben, die den angegebenen Kapazitätsanbieter verwenden. Betrachten Sie das folgende Beispiel. Sie haben eine Strategie, die zwei Kapazitätsanbieter enthält, und beide haben eine Gewichtung von 1. Wenn der Basisprozentsatz erreicht ist, werden die Aufgaben gleichmäßig auf die beiden Kapazitätsanbieter aufgeteilt. Mit der gleichen Logik können Sie für capacityProviderA eine Gewichtung von 1 und für capacityProviderB eine Gewichtung von 4 festlegen. Dann gibt es für jede Aufgabe, die mit capacityProviderA ausgeführt wird, vier Aufgaben, die capacityProviderB verwenden.

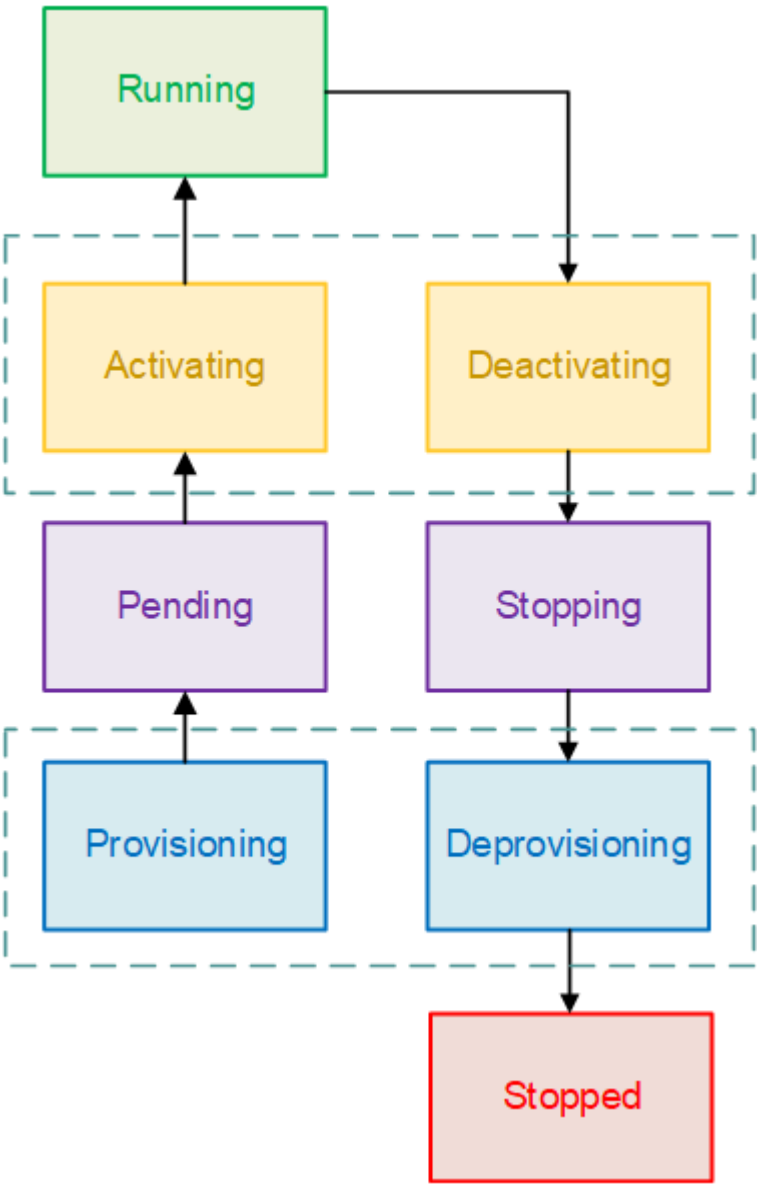
Der Starttyp startet Ihre Aufgaben direkt auf Fargate oder auf den Amazon EC2 EC2-Instances, die Sie manuell in Ihren Clustern registriert haben.

Amazon ECS-Aufgabenlebenszyklus

Wenn eine Aufgabe entweder manuell oder als Teil eines Service gestartet wird, kann sie mehrere Zustände durchlaufen, bevor sie automatisch endet oder manuell angehalten wird. Einige Aufgaben sollen als Stapelverarbeitungsaufträge ausgeführt werden, die vom Zustand `PENDING` zu `RUNNING` zu `STOPPED` fortschreiten. Andere Aufgaben, die Teil eines Service sein können, sollen immer weiter ausgeführt werden oder nach Bedarf herauf- oder herunterskaliert werden.

Wenn Änderungen am Aufgabenstatus angefragt werden, wie etwa das Beenden einer Aufgabe oder das Aktualisieren der gewünschten Anzahl eines Service, um ihn hoch- oder herunterzuskalieren, speichert der Amazon-ECS-Container-Agent diese Änderungen als den letzten bekannten Status (`lastStatus`) der Aufgabe und den gewünschten Status (`desiredStatus`) der Aufgabe. Sowohl der letzte bekannte Status als auch der gewünschte Status einer Aufgabe kann entweder in der Konsole oder durch Beschreiben der Aufgabe mit der API oder der AWS CLI angezeigt werden.

Das Flussdiagramm unten zeigt den Lebenszyklusfluss von Aufgaben.



Lebenszyklusstati

Es folgen Beschreibungen der einzelnen Lebenszyklusstati von Aufgaben.

PROVISIONING

Amazon ECS muss zusätzliche Schritte durchführen, bevor die Aufgabe gestartet wird. Beispiel: Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, muss die Elastic-Network-Schnittstelle bereitgestellt werden.

PENDING

Hierbei handelt es sich um einen Übergangsstatus, in dem Amazon ECS darauf wartet, dass der Container-Agent weitere Maßnahmen ergreift. Eine Aufgabe hat den Zustand „Pending“ (Ausstehend), bis Ressourcen für die Aufgabe verfügbar sind.

ACTIVATING

Dies ist ein Übergangszustand, in dem Amazon ECS zusätzliche Schritte durchführen muss, nachdem die Aufgabe gestartet wurde, aber bevor die Aufgabe in den Zustand `RUNNING` übergehen kann. Beispiel: Für Aufgaben, für die Service Discovery konfiguriert ist, müssen Service Discovery-Ressourcen erstellt werden. Für Aufgaben, die Teil eines Service sind, der so konfiguriert ist, dass er mehrere Zielgruppen für das Elastic Load Balancing verwendet, erfolgt die Zielgruppenregistrierung in diesem Zustand.

AUSFÜHREN

Die Aufgabe wird erfolgreich ausgeführt.

DEACTIVATING

Dies ist ein Übergangsstatus, in dem Amazon ECS zusätzliche Schritte durchführen muss, bevor die Aufgabe gestoppt wird. Beispiel: Für Aufgaben, die Teil eines Service sind, der so konfiguriert ist, dass er mehrere Zielgruppen für das Elastic Load Balancing verwendet, erfolgt die Aufhebung der Zielgruppenregistrierung in diesem Zustand.

STOPPING

Hierbei handelt es sich um einen Übergangsstatus, in dem Amazon ECS darauf wartet, dass der Container-Agent weitere Maßnahmen ergreift.

Bei Linux-Containern sendet der Container-Agent das `SIGTERM` Signal, um zu benachrichtigen, dass die Anwendung beendet und heruntergefahren werden muss, und sendet dann `SIGKILL` nach Ablauf der in der Aufgabendefinition festgelegten Wartezeit `StopTimeout` ein.

DEPROVISIONING

Amazon ECS muss zusätzliche Schritte durchführen, nachdem die Aufgabe gestoppt wurde und bevor sie in den Status `STOPPED` übergeht. Beispiel: Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, muss die Elastic-Network-Schnittstelle getrennt und gelöscht werden.

STOPPED

Die Aufgabe wurde erfolgreich gestoppt.

Wenn Ihre Aufgabe aufgrund eines Fehlers beendet wurde, finden Sie weitere Informationen unterFehler beim Beenden von Amazon ECS-Aufgaben anzeigen.

GELÖSCHT

Dies ist ein Übergangszustand, wenn eine Aufgabe angehalten wird. Dieser Status wird nicht in der Konsole angezeigt, aber in `describe-tasks`.

Wie Amazon ECS Aufgaben auf Container-Instances platziert

Mithilfe von Task Placement können Sie Amazon ECS so konfigurieren, dass Ihre Aufgaben auf Container-Instances platziert werden, die bestimmte Kriterien erfüllen, z. B. eine Availability Zone oder einen Instance-Typ.

Im Folgenden sind die Komponenten für die Aufgabenverteilung aufgeführt:

- Strategie zur Aufgabenplatzierung — Der Algorithmus zur Auswahl von Container-Instances für die Aufgabenplatzierung oder von Aufgaben für die Beendigung. Amazon ECS kann Container-Instances beispielsweise nach dem Zufallsprinzip auswählen, oder es kann Container-Instances so auswählen, dass die Aufgaben gleichmäßig auf eine Gruppe von Instances verteilt werden.
- Aufgabengruppe — Eine Gruppe verwandter Aufgaben, zum Beispiel Datenbankaufgaben.
- Beschränkung der Aufgabenplatzierung — Dies sind Regeln, die erfüllt sein müssen, um eine Aufgabe auf einer Container-Instance platzieren zu können. Wenn die Einschränkung nicht erfüllt ist, wird die Aufgabe nicht platziert und verbleibt im `PENDING` Status. Sie können beispielsweise eine Einschränkung verwenden, um Aufgaben nur einem bestimmten Instanztyp zuzuweisen.

Amazon ECS hat unterschiedliche Algorithmen für die Starttypen.

EC2-Starttyp

Für Aufgaben, die den EC2-Starttyp verwenden, muss Amazon ECS anhand der in der Aufgabendefinition angegebenen Anforderungen, wie CPU und Arbeitsspeicher, bestimmen, wo die Aufgabe platziert werden soll. Wenn Sie die Anzahl der Aufgaben herunterskalieren, muss Amazon ECS auf ähnliche Weise bestimmen, welche Aufgaben beendet werden sollen. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie festlegen, wie Amazon ECS Aufgaben platziert und beendet.

Die Standardstrategien für die Aufgabenplatzierung hängen davon ab, ob Sie Aufgaben manuell (eigenständige Aufgaben) oder innerhalb eines Service ausführen. Für Aufgaben, die als Teil eines Amazon-ECS-Service ausgeführt werden, ist die Strategie zur Aufgabenplatzierung `spread` unter Verwendung von `attribute:ecs.availability-zone`. Es gibt keine Standardbeschränkung für die Aufgabenplatzierung für Aufgaben in Services. Weitere Informationen finden Sie unter Planen Sie Ihre Container auf Amazon ECS.

> **Note**
>
> Aufgabenplatzierungsstrategien entsprechen bestem Bemühen. Amazon ECS versucht auch dann noch, Aufgaben zu platzieren, wenn die optimale Platzierungsoption nicht verfügbar ist. Einschränkungen der Aufgabenplatzierung sind jedoch verbindlich und können eine Aufgabenplatzierung verhindern.

Sie können Aufgabenplatzierungsstrategien mit Bedingungen kombinieren. Beispielsweise können Sie eine Aufgabenplatzierungsstrategie und eine Aufgabenplatzierungsbeschränkung verwenden, um Aufgaben auf Availability Zones zu verteilen und Bin-Pack-Aufgaben basierend auf dem Arbeitsspeicher innerhalb jeder Availability Zone zu verteilen (jedoch nur für G2-Instances).

Wenn Amazon ECS Aufgaben platziert, verwendet es das folgende Verfahren zum Auswählen von Container-Instances:

1. Identifizieren Sie die Container-Instances, die die CPU-, GPU-, Arbeitsspeicher- und Port-Anforderungen in der Aufgabendefinition erfüllen.
2. Identifizieren Sie die Container-Instances, die die Einschränkungen bei der Aufgabenplatzierung erfüllen.
3. Identifizieren Sie die Container-Instances, die die Strategien zur Aufgabenplatzierung erfüllen.
4. Wählen Sie die Container-Instances für die Aufgabenplatzierung aus.

Fargate Starttyp

Aufgabenplatzierungs-Strategien und -Beschränkungen werden für Aufgaben mit dem Fargate-Launchtyp nicht unterstützt. Fargate wird sich bemühen, Aufgaben über zugängliche Availability Zones zu verteilen. Wenn der Kapazitätsanbieter sowohl Fargate als auch Fargate Spot umfasst, ist das Verteilungsverhalten für jeden Kapazitätsanbieter unabhängig.

Verwenden Sie Strategien, um die Amazon ECS-Aufgabenverteilung zu definieren

Für Aufgaben, die den EC2-Starttyp verwenden, muss Amazon ECS anhand der in der Aufgabendefinition angegebenen Anforderungen, wie CPU und Arbeitsspeicher, bestimmen, wo die Aufgabe platziert werden soll. Wenn Sie die Anzahl der Aufgaben herunterskalieren, muss Amazon ECS auf ähnliche Weise bestimmen, welche Aufgaben beendet werden sollen. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie festlegen, wie Amazon ECS Aufgaben platziert und beendet.

Die Standardstrategien für die Aufgabenplatzierung hängen davon ab, ob Sie Aufgaben manuell (eigenständige Aufgaben) oder innerhalb eines Service ausführen. Für Aufgaben, die als Teil eines Amazon-ECS-Service ausgeführt werden, ist die Strategie zur Aufgabenplatzierung `spread` unter Verwendung von `attribute:ecs.availability-zone`. Es gibt keine Standardbeschränkung für die Aufgabenplatzierung für Aufgaben in Services. Weitere Informationen finden Sie unter Planen Sie Ihre Container auf Amazon ECS.

> **Note**
>
> Aufgabenplatzierungsstrategien entsprechen bestem Bemühen. Amazon ECS versucht auch dann noch, Aufgaben zu platzieren, wenn die optimale Platzierungsoption nicht verfügbar ist. Einschränkungen der Aufgabenplatzierung sind jedoch verbindlich und können eine Aufgabenplatzierung verhindern.

Sie können Aufgabenplatzierungsstrategien mit Bedingungen kombinieren. Beispielsweise können Sie eine Aufgabenplatzierungsstrategie und eine Aufgabenplatzierungsbeschränkung verwenden, um Aufgaben auf Availability Zones zu verteilen und Bin-Pack-Aufgaben basierend auf dem Arbeitsspeicher innerhalb jeder Availability Zone zu verteilen (jedoch nur für G2-Instances).

Wenn Amazon ECS Aufgaben platziert, verwendet es das folgende Verfahren zum Auswählen von Container-Instances:

1. Identifizieren Sie die Container-Instances, die die CPU-, GPU-, Arbeitsspeicher- und Port-Anforderungen in der Aufgabendefinition erfüllen.
2. Identifizieren Sie die Container-Instances, die die Einschränkungen bei der Aufgabenplatzierung erfüllen.

3. Identifizieren Sie die Container-Instances, die die Strategien zur Aufgabenplatzierung erfüllen.
4. Wählen Sie die Container-Instances für die Aufgabenplatzierung aus.

Sie geben Strategien zur Aufgabenplatzierung in der Servicedefinition oder der Aufgabendefinition mithilfe des `placementStrategy` Parameters an.

```
"placementStrategy": [
    {
        "field": "The field to apply the placement strategy against",
        "type": "The placement strategy to use"
    }
]
```

Sie können die Strategien angeben, wenn Sie eine Aufgabe ausführen (RunTask), einen neuen Dienst erstellen (CreateService) oder einen vorhandenen Dienst aktualisieren (UpdateService).

In der folgenden Tabelle werden die verfügbaren Typen und Felder beschrieben.

Typ	Gültige Feldwerte	
`binpack` Aufgaben werden auf Container-Instances platziert, um so wenig ungenutzten CPU- oder Arbeitsspeicherplatz wie möglich zu belassen. Diese Strategie minimiert die Anzahl der verwendeten Container-Instances. Wenn diese Strategie verwendet wird und eine Abskalierungs-Aktion durchgeführt wird, beendet Amazon ECS Aufgaben. Es tut dies basierend auf der Menge der Ressourcen, die nach dem	• cpu • memory	

Typ	Gültige Feldwerte	
Beenden der Aufgabe in der Container-Instance verbleiben. Die Container-Instance, die nach der Beendigung der Aufgabe die meisten verfügbaren Ressourcen hat, veranlasst die Beendigung dieser Aufgabe.		
`random` Aufgaben werden nach dem Zufallsprinzip platziert.	Nicht verwendet	

Typ	Gültige Feldwerte	
`spread` Aufgaben werden gleichmäßig basierend auf dem angegebenen Wert platziert. Service-Aufgaben werden basierend auf den Aufgaben von diesem Service verteilt. Eigenständige Aufgaben werden basierend auf den Aufgaben derselben Aufgabengruppe verteilt. Weitere Informationen über Aufgabengruppen finden Sie unter Gruppenbezogene Amazon ECS-Aufgaben . Wenn die `spread`-Strategie verwendet wird und eine Abskalierungs-Aktion durchgeführt wird, wählt Amazon ECS Aufgaben zum Beenden aus, die ein Gleichgewicht über Availability Zones aufrechterhalten. Innerhalb einer Availability Zone werden Aufgaben nach dem Zufallsprinzip ausgewählt.	• `instanceId` (oder`host`, was den gleichen Effekt hat) • jede Plattform oder jedes benutzerdefinierte Attribut, das auf eine Container-Instance angewendet wird, wie `attribute:ecs.availability-zone`	

Die Aufgabenplatzierungsstrategien können auch für bestehende Services aktualisiert werden. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

Sie können eine Strategie zur Aufgabenverteilung erstellen, die mehrere Strategien verwendet, indem Sie Arrays von Strategien in der Reihenfolge erstellen, in der sie ausgeführt werden sollen. Wenn

Sie beispielsweise Aufgaben auf Availability Zones und anschließend mit der Binpacks-Strategie basierend auf dem Speicher innerhalb der einzelnen Availability Zone platzieren möchten, geben Sie die Availability-Zone-Strategie gefolgt von der Speicherstrategie an. Beispielstrategien finden Sie unter Beispiele für Amazon ECS-Strategien zur Aufgabenplatzierung.

Beispiele für Amazon ECS-Strategien zur Aufgabenplatzierung

Sie können Strategien zur Aufgabenplatzierung mit den folgenden Aktionen festlegen: CreateServiceUpdateService, und RunTask.

Beispiele

- Gleichmäßiges Verteilen von Aufgaben auf Availability Zones
- Gleichmäßiges Verteilen von Aufgaben auf alle Instances
- Binpacks-Aufgaben basierend auf dem Speicher
- Zufälliges Platzieren von Aufgaben
- Verteilen Sie die Aufgaben gleichmäßig über die Availability Zones und verteilen Sie sie dann gleichmäßig auf die Instances innerhalb jeder Availability Zone
- Verteilen Sie Aufgaben gleichmäßig auf Availability Zones und anschließend mit der Binpack-Strategie basierend auf dem Speicher innerhalb jeder Availability Zone
- Gleichmäßige Verteilung der Aufgaben auf die Instances, woraufhin Aufgaben auf der Grundlage des Speichers zu Bin-Packs zugeordnet werden

Gleichmäßiges Verteilen von Aufgaben auf Availability Zones

Mit der folgenden Strategie werden Aufgaben gleichmäßig auf Availability Zones verteilt.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
]
```

Gleichmäßiges Verteilen von Aufgaben auf alle Instances

Mit der folgenden Strategie werden Aufgaben gleichmäßig auf alle Instances verteilt.

```
"placementStrategy": [
```

```
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Binpacks-Aufgaben basierend auf dem Speicher

Mit der folgenden Strategie werden Binpack-Aufgaben basierend auf dem Speicherplatz erstellt.

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Zufälliges Platzieren von Aufgaben

Mit der folgenden Strategie werden Aufgaben zufällig platziert.

```
"placementStrategy": [
    {
        "type": "random"
    }
]
```

Verteilen Sie die Aufgaben gleichmäßig über die Availability Zones und verteilen Sie sie dann gleichmäßig auf die Instances innerhalb jeder Availability Zone

Mit der folgenden Strategie werden Aufgaben gleichmäßig auf Availability Zones und anschließend gleichmäßig auf die Instances innerhalb der einzelnen Availability Zone verteilt.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "instanceId",
        "type": "spread"
    }
```

```
]
```

Verteilen Sie Aufgaben gleichmäßig auf Availability Zones und anschließend mit der Binpack-Strategie basierend auf dem Speicher innerhalb jeder Availability Zone

Mit der folgenden Strategie werden Aufgaben gleichmäßig auf Availability Zones und anschließend mit der Binpack-Strategie basierend auf dem Speicher innerhalb der einzelnen Availability Zone platziert.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Gleichmäßige Verteilung der Aufgaben auf die Instances, woraufhin Aufgaben auf der Grundlage des Speichers zu Bin-Packs zugeordnet werden

Mit der folgenden Strategie werden Aufgaben gleichmäßig auf alle Instances und anschließend mit der Binpacks-Strategie basierend auf dem Speicher innerhalb der einzelnen Instance platziert.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Gruppenbezogene Amazon ECS-Aufgaben

Sie können eine Reihe verwandter Aufgaben identifizieren und sie einer Aufgabengruppe zuordnen. Alle Aufgaben mit demselben Aufgabengruppennamen werden bei der Aufgabenplatzierungsstrategie

mit `spread` als Satz betrachtet. Nehmen wir zum Beispiel an, dass Sie verschiedene Anwendungen in einem Cluster ausführen, beispielsweise Datenbanken und Webserver. Um sicherzustellen, dass Ihre Datenbanken gleichmäßig auf die Availability Zones verteilt sind, fügen Sie sie zu einer Aufgabengruppe mit der Bezeichnung `databases` hinzu und verwenden Sie anschließend die `spread`-Aufgabenplatzierungsstrategie. Weitere Informationen finden Sie unter Verwenden Sie Strategien, um die Amazon ECS-Aufgabenverteilung zu definieren.

Aufgabengruppen können auch als Einschränkung für die Aufgabenplatzierung verwendet werden. Wenn Sie eine Aufgabengruppe in der `memberOf`-Einschränkung angeben, werden die Aufgaben nur an Container-Instances gesendet, die sich in der angegebenen Aufgabengruppe befinden. Ein Beispiel finden Sie unter Beispiel für Einschränkungen bei der Aufgabenplatzierung in Amazon ECS.

Standardmäßig verwenden eigenständige Aufgaben den Familiennamen der Aufgabendefinition (z. B. `family:my-task-definition`) als Aufgabengruppen-Namen, wenn kein benutzerdefinierter Aufgabengruppen-Name angegeben ist. Aufgaben, die als Teil eines Dienstes gestartet werden, verwenden den Dienstnamen als Aufgabengruppenname und können nicht geändert werden.

Für die Aufgabengruppe gelten die folgenden Anforderungen.

- Ein Aufgabengruppen-Name darf höchstens 255 Zeichen lang sein.
- Jede Aufgabe kann nur einer Gruppe zugeordnet werden.
- Nach dem Launchen einer Aufgabe können Sie deren Aufgabengruppe nicht modifizieren.

Definieren Sie, welche Container-Instances Amazon ECS für Aufgaben verwendet

Eine Aufgabenplatzierungsbeschränkung ist eine Regel für eine Container-Instance, anhand derer Amazon ECS bestimmt, ob die Aufgabe auf der Instance ausgeführt werden darf. Mindestens eine Container-Instance muss der Einschränkung entsprechen. Wenn es keine Instances gibt, die der Einschränkung entsprechen, verbleibt die Aufgabe im Status `PENDING`. Wenn Sie einen neuen Service erstellen oder einen vorhandenen aktualisieren, können Sie Einschränkungen bei der Aufgabenplatzierung für die Aufgaben des Services angeben.

Mithilfe des `placementConstraint` Parameters können Sie Einschränkungen für die Aufgabenplatzierung in der Servicedefinition, Aufgabendefinition oder Aufgabe angeben.

```
"placementConstraint": [
    {
```

```
        "expression": "The expression that defines the task placement constraints",
        "type": "The placement constraint type to use"
    }
]
```

In der folgenden Tabelle wird beschrieben, wie die Parameter verwendet werden.

Einschränkungstypen	Kann angegeben werden, wann	
`distinctInstance` Platzierung jeder Aufgabe auf einer anderen Container-Instance. **Important** Wir empfehlen Kunden, die für ihre Aufgaben eine starke Isolierung suchen, Fargate zu verwenden . Fargate führt jede Aufgabe in einer Hardware-Virtualis ierungsumgebung aus. Dadurch wird sichergestellt, dass sich diese container isierten Workloads keine Netzwerks chnittstellen, kurzlebig en Fargate-Speicher, CPU oder Arbeitssp eicher mit anderen Aufgaben teilen. Weitere Informati	• Eine Aufgabe ausführen RunTask • Einen neuen Dienst erstellen CreateService,	

Einschränkungstypen	Kann angegeben werden, wann	
onen finden Sie unter Sicherheitsüberblick von. AWS Fargate		
`memberOf` Platzierung von Aufgaben auf Container-Instances, die einem Ausdruck entsprechen.	• Eine Aufgabe ausführen RunTask • Einen neuen Dienst erstellen CreateService, • Eine neue RegisterTaskAufgabendefinitionsdefinition erstellen • Eine neue Version einer RegisterTaskAufgabendefinitionsdefinition erstellen • Einen Dienst aktualisieren UpdateService	

Wenn Sie den `memberOf` Einschränkungstyp verwenden, können Sie mithilfe der Cluster-Abfragesprache einen Ausdruck erstellen, der die Container-Instances definiert, in denen Amazon ECS Aufgaben platzieren kann. Der Ausdruck bietet Ihnen die Möglichkeit, Ihre Container-Instances nach Attributen zu gruppieren. Der Ausdruck gehört in den `expression` Parameter von`placementConstraint`.

Attribute der Amazon ECS-Container-Instance

Sie können zu Ihren Container-Instances benutzerdefinierte Metadaten hinzufügen, die als Attribute bezeichnet werden. Jedes Attribut hat einen Namen und einen optionalen Zeichenfolgenwert. Sie können die von Amazon ECS vergebenen Attribute verwenden oder benutzerdefinierte Attribute erstellen.

Die folgenden Abschnitte enthalten Beispiele für integrierte, optionale und benutzerdefinierte Attribute.

Integrierte Attribute

Amazon ECS wendet automatisch die folgenden Attribute auf Ihre Container-Instances an.

`ecs.ami-id`

Die ID des zum Start der Instance verwendeten AMI. Ein Beispielwert für dieses Attribut ist `ami-1234abcd`.

`ecs.availability-zone`

Die Availability Zone für die Instance. Ein Beispielwert für dieses Attribut ist `us-east-1a`.

`ecs.instance-type`

Den Instance-Typ für die Instance. Ein Beispielwert für dieses Attribut ist `g2.2xlarge`.

`ecs.os-type`

Das Betriebssystem für die Instance. Die möglichen Werte für dieses Attribut sind `linux` und `windows`.

`ecs.os-family`

Die Betriebssystemversion für die Instance.

Für Linux-Instances ist der gültige Wert `LINUX`. Für Windows-Instances legt ECS den Wert im Format `WINDOWS_SERVER_<OS_Release>_<FULL or CORE>` fest. Die gültigen Werte sind `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_20H2_CORE`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE` und `WINDOWS_SERVER_2016_FULL`.

Dies ist wichtig für Windows-Container und Windows containers on AWS Fargate weil die Betriebssystemversion jedes Windows-Containers mit der des Hosts übereinstimmen muss. Wenn sich die Windows-Version des Container-Images von der des Hosts unterscheidet, startet der Container nicht. Weitere Informationen finden Sie unter Kompatibilität mit Windows-Containern auf der Microsoft-Dokumentations-Website.

Wenn auf Ihrem Cluster mehrere Windows-Versionen ausgeführt werden, können Sie mithilfe der Platzierungsbeschränkung `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)` sicherstellen, dass eine Aufgabe auf einer EC2-Instance platziert wird, die auf derselben Version ausgeführt wird. Weitere

Informationen finden Sie unter the section called “Amazon ECS-optimierte Windows AMI-Metadaten abrufen”.

`ecs.cpu-architecture`

Die CPU-Architektur für die Instance. Beispielwerte für dieses Attribut sind `x86_64` und `arm64`.

`ecs.vpc-id`

Die VPC, in der die Instance gestartet wurde. Ein Beispielwert für dieses Attribut ist `vpc-1234abcd`.

`ecs.subnet-id`

Das Subnetz, das die Instance verwendet. Ein Beispielwert für dieses Attribut ist `subnet-1234abcd`.

Optionale Attribute

Amazon ECS kann die folgenden Attribute zu Ihren Container-Instances hinzufügen.

`ecs.awsvpc-trunk-id`

Wenn dieses Attribut vorhanden ist, verfügt die Instance über eine Trunk-Netzwerkschnittstelle. Weitere Informationen finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.

`ecs.outpost-arn`

Wenn dieses Attribut existiert, enthält es den Amazon-Ressourcennamen (ARN) des Outpost. Weitere Informationen finden Sie unter the section called “Amazon Elastic Container Service auf AWS Outposts”.

`ecs.capability.external`

Wenn dieses Attribut vorhanden ist, wird die Instance als externe Instance identifiziert. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den externen Starttyp.

Custom attributes (Benutzerdefinierte Attribute)

Sie können benutzerdefinierte Attribute für Ihre Container-Instances festlegen. Beispielsweise können Sie ein Attribut mit dem Namen „stack“ und dem Wert „prod“ definieren.

Bei der Angabe benutzerdefinierter Attribute sollte Folgendes berücksichtigt werden.

- `name` muss zwischen 1 und 128 Zeichen sein und Name kann Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche, Unterstriche, Schrägstriche, Gegenschrägstriche oder Punkte enthalten.
- `value` muss zwischen 1 und 128 Zeichen sein und Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche, Unterstriche, Punkte, At-Zeichen (@), Schrägstriche, Gegenschrägstriche, Doppelpunkte oder Leerzeichen enthalten. Der Wert darf keine führenden oder nachgestellten Leerzeichen enthalten.

Erstellen Sie Ausdrücke, um Container-Instances für Amazon ECS-Aufgaben zu definieren

Cluster-Abfragen sind Ausdrücke, die Ihnen das Gruppieren von Objekten ermöglichen. Beispielsweise können Sie Container-Instances nach Attributen gruppieren, z. B. Availability Zone, Instance-Typ oder benutzerdefinierte Metadaten. Weitere Informationen finden Sie unter Attribute der Amazon ECS-Container-Instance.

Nachdem Sie eine Gruppe von Container-Instances definiert haben, können Sie Amazon ECS so anpassen, dass Aufgaben basierend auf der Gruppe auf Container-Instances platziert werden. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen und Einen Amazon ECS-Service mithilfe der Konsole erstellen. Sie können bei der Auflistung von Container-Instances auch einen Gruppenfilter anwenden.

Ausdruck-Syntax

Ausdrücke haben die folgende Syntax:

```
subject operator [argument]
```

Betreff

Das auszuwertende Attribut oder Feld.

`agentConnected`

Wählen Sie Container-Instances anhand des Verbindungsstatus ihres Amazon-ECS-Container-Agenten aus. Mithilfe dieses Filters können Sie nach Instances mit getrennten Container-Agenten suchen.

Gültige Operatoren: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`agentVersion`

Wählen Sie Container-Instances anhand der Version ihres Amazon-ECS-Container-Agenten aus. Mithilfe dieses Filters können Sie Instances suchen, auf denen veraltete Versionen des Amazon-ECS-Container-Agenten ausgeführt werden.

Gültige Operatoren: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`attribute:`*`attribute-name`*

Wählen Sie Container-Instances nach Attribut aus. Weitere Informationen finden Sie unter Attribute der Amazon ECS-Container-Instance.

`ec2InstanceId`

Wählen Sie Container-Instances nach ihrer Amazon-EC2-Instance-ID aus.

Gültige Operatoren: equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`registeredAt`

Wählen Sie Container-Instances nach ihrem Registrierungsdatum aus. Mithilfe dieses Filters können Sie neu registrierte Instances oder sehr alte Instances finden.

Gültige Operatoren: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

Gültige Datumsformate: 2018-06-18T22:28:28+00:00, 2018-06-18T22:28:28Z, 2018-06-18T22:28:28, 2018-06-18

`runningTasksCount`

Wählen Sie Container-Instances nach der Anzahl der ausgeführten Aufgaben aus. Mithilfe dieses Filters können Sie Instances finden, die leer oder nahezu leer sind, auf denen also wenige Aufgaben ausgeführt werden.

Gültige Operatoren: equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`task:group`

Wählen Sie Container-Instances nach Aufgabengruppe aus. Weitere Informationen finden Sie unter Gruppenbezogene Amazon ECS-Aufgaben .

Operator

Der Vergleichsoperator. Folgende Operatoren werden unterstützt.

Operator	Beschreibung
==, equals	Zeichenfolgen-Übereinstimmung
!=, not_equals	Keine Zeichenfolgen-Übereinstimmung
>, greater_than	größer als
>=, greater_than_equal	größer als oder gleich
<, less_than	kleiner als
<=, less_than_equal	kleiner als oder gleich
exists	Subjekt ist vorhanden
!exists, not_exists	Subjekt ist nicht vorhanden
in	Wert in Argumentliste enthalten
!in, not_in	Wert nicht in Argumentliste enthalten
=~, matches	Muster-Übereinstimmung
!~, not_matches	Keine Muster-Übereinstimmung

> **Note**
>
> Ein einzelner Ausdruck kann keine Klammern enthalten. Klammern können jedoch in zusammengesetzten Ausdrücken verwendet werden, um eine Rangfolge anzugeben.

Argument

Bei vielen Operatoren ist das Argument ein Literalwert.

Die Operatoren `in` und `not_in` setzen eine Argumentliste als Argument voraus. Geben Sie eine Argumentliste wie folgt an:

```
[argument1, argument2, ..., argumentN]
```

Die Operatoren „matches“ und „not_matches“ setzen ein Argument voraus, das die reguläre Java-Ausdrucksyntax erfüllt. Weitere Informationen finden Sie unter java.util.regex.Pattern.

Zusammengesetzte Ausdrücke

Sie können Ausdrücke mit den folgenden booleschen Operatoren miteinander kombinieren:

- &&, und
- ||, oder
- !, nicht

Sie können Klammern verwenden, um eine Rangfolge anzugeben:

```
(expression1 or expression2) and expression3
```

Beispiel-Ausdrücke

Es folgen Beispiel-Ausdrücke.

Beispiel: Zeichenfolgen-Übereinstimmung

Mit dem folgenden Ausdruck werden Instances des angegebenen Instance-Typs ausgewählt.

```
attribute:ecs.instance-type == t2.small
```

Beispiel: Argumentliste

Mit dem folgenden Ausdruck werden Instances in der Availability Zone us-east-1a oder us-east-1b ausgewählt.

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

Beispiel: zusammengesetzter Ausdruck

Mit dem folgenden Ausdruck werden G2-Instances ausgewählt, die nicht in der Availability Zone us-east-1d enthalten sind.

```
attribute:ecs.instance-type =~ g2.* and attribute:ecs.availability-zone != us-east-1d
```

Beispiel: Aufgaben-Affinität

Mit dem folgenden Ausdruck werden Instances ausgewählt, die Aufgaben in der Gruppe `service:production` hosten.

```
task:group == service:production
```

Beispiel: Aufgaben-Anti-Affinität

Mit dem folgenden Ausdruck werden Instances ausgewählt, die keine Aufgaben in der Datenbankgruppe hosten.

```
not(task:group == database)
```

Beispiel: Anzahl der ausgeführten Aufgaben

Mit dem folgenden Ausdruck werden Instances ausgewählt, auf denen nur eine Aufgabe ausgeführt wird.

```
runningTasksCount == 1
```

Beispiel: Amazon-ECS-Container-Agent-Version

Mit dem folgenden Ausdruck werden Instances ausgewählt, auf denen eine Container-Agenten-Version unter Version 1.14.5 ausgeführt wird.

```
agentVersion < 1.14.5
```

Beispiel: Zeitpunkt der Instance-Registrierung

Mit dem folgenden Ausdruck werden Instances ausgewählt, die vor dem 13. Februar 2018 registriert wurden.

```
registeredAt < 2018-02-13
```

Beispiel: ID der Amazon-EC2-Instance

Mit dem folgenden Ausdruck werden Instances mit den folgenden Amazon-EC2-Instance-IDs ausgewählt.

```
ec2InstanceId in ['i-abcd1234', 'i-wxyx7890']
```

Beispiel für Einschränkungen bei der Aufgabenplatzierung in Amazon ECS

Nachfolgend finden Sie Beispiele für Aufgabenplatzierungsbedingungen.

In diesem Beispiel wird die `memberOf` Einschränkung verwendet, um Aufgaben auf T2-Instances zu platzieren. Sie kann mit den folgenden Aktionen angegeben werden: CreateServiceUpdateService,, RegisterTaskDefinition und RunTask.

```
"placementConstraints": [
    {
        "expression": "attribute:ecs.instance-type =~ t2.*",
        "type": "memberOf"
    }
]
```

Das Beispiel verwendet die Einschränkung `memberOf`, um Replikataufgaben auf Instances mit Aufgaben in der `daemon-service`-Aufgabengruppe des Daemon-Service zu platzieren, wobei alle ebenfalls angegebenen Strategien zur Aufgabenplatzierung berücksichtigt werden. Diese Einschränkung stellt sicher, dass die Daemon-Serviceaufgaben vor den Replikatserviceaufgaben auf der EC2-Instance platziert werden.

Ersetzen Sie `daemon-service` durch den Namen des Daemon-Services.

```
"placementConstraints": [
    {
        "expression": "task:group == service:daemon-service",
        "type": "memberOf"
    }
]
```

In dem Beispiel wird die `memberOf`-Bedingung verwendet, um Aufgaben in Instances mit anderen Aufgaben in der `databases`-Aufgabengruppe unter Beachtung aller ebenfalls angegebenen Strategien zur Aufgaben-Platzierung zu platzieren. Weitere Informationen über Aufgabengruppen

finden Sie unter Gruppenbezogene Amazon ECS-Aufgaben . Es kann mit den folgenden Aktionen angegeben werden: CreateServiceUpdateService,, RegisterTaskDefinition und RunTask.

```
"placementConstraints": [
    {
        "expression": "task:group == databases",
        "type": "memberOf"
    }
]
```

Mit der Bedingung `distinctInstance` wird jede Aufgabe in der Gruppe auf einer anderen Instance platziert. Es kann mit den folgenden Aktionen angegeben werden: CreateService, UpdateService, und RunTask

```
"placementConstraints": [
    {
        "type": "distinctInstance"
    }
]
```

Eigenständige Amazon ECS-Aufgaben

Sie können Ihre Anwendung als Aufgabe ausführen, wenn Sie eine Anwendung haben, die bestimmte Aufgaben ausführt und dann stoppt, z. B. einen Batch-Prozess. Wenn Sie eine Aufgabe einmal ausführen möchten, können Sie die Konsole AWS CLI, APIs oder SDKs verwenden.

Wenn Sie Ihre Anwendung nach einem ratenbasierten, cron-basierten oder einmaligen Zeitplan ausführen müssen, können Sie mit Scheduler einen Zeitplan erstellen. EventBridge

Arbeitsablauf für Aufgaben

Wenn Sie Amazon ECS-Aufgaben starten (eigenständige Aufgaben oder durch Amazon ECS-Services), wird eine Aufgabe erstellt und zunächst in den `PROVISIONING` Status verschoben. Wenn sich eine Aufgabe im `PROVISIONING` Status befindet, sind weder die Aufgabe noch die Container vorhanden, da Amazon ECS Rechenkapazität für die Platzierung der Aufgabe finden muss.

Amazon ECS wählt die passende Rechenkapazität für Ihre Aufgabe auf der Grundlage Ihres Starttyps oder der Konfiguration Ihres Kapazitätsanbieters aus. Sie können Kapazitätsanbieter und Kapazitätsanbieterstrategien sowohl für die Starttypen Fargate als auch Amazon EC2 verwenden. Mit Fargate müssen Sie sich keine Gedanken über die Bereitstellung, Konfiguration und Skalierung

Ihrer Clusterkapazität machen. Fargate kümmert sich um das gesamte Infrastrukturmanagement für Ihre Aufgaben. Für den EC2-Starttyp können Sie entweder Ihre Clusterkapazität verwalten, indem Sie Amazon EC2 EC2-Instances in Ihrem Cluster registrieren, oder Sie können Cluster Auto Scaling verwenden, um Ihre Rechenkapazitätsverwaltung zu vereinfachen. Cluster Auto Scaling sorgt für die dynamische Skalierung Ihrer Clusterkapazität, sodass Sie sich auf die Ausführung von Aufgaben konzentrieren können. Amazon ECS bestimmt anhand der Anforderungen, die Sie in der Aufgabendefinition angeben, wie CPU und Arbeitsspeicher, sowie anhand Ihrer Platzierungsbeschränkungen und -strategien, wo die Aufgabe platziert werden soll. Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

Wenn Sie einen Kapazitätsanbieter mit aktivierter verwalteter Skalierung verwenden, werden Aufgaben, die aufgrund mangelnder Rechenkapazität nicht gestartet werden können, in den `PROVISIONING` Status versetzt, anstatt sofort fehlzuschlagen. Nachdem die Kapazität für die Platzierung Ihrer Aufgabe ermittelt wurde, stellt Amazon ECS die erforderlichen Anlagen bereit (z. B. Elastic Network Interfaces (ENIs) für Aufgaben im `awsvpc` Modus). Es verwendet den Amazon ECS-Container-Agenten, um Ihre Container-Images abzurufen und dann Ihre Container zu starten. Nachdem die Bereitstellung abgeschlossen und die entsprechenden Container gestartet wurden, versetzt Amazon ECS die Aufgabe in den `RUNNING` Status. Informationen zu den Aufgabenstatus finden Sie unterAmazon ECS-Aufgabenlebenszyklus.

Optimieren Sie die Startzeit von Amazon ECS-Aufgaben

Beachten Sie die folgenden Empfehlungen, um den Start Ihrer Aufgaben zu beschleunigen.

- Container-Images und Binpack-Instances zwischenspeichern

  Wenn Sie den EC2-Starttyp verwenden, können Sie das Pull-Verhalten des Amazon ECS-Container-Agenten wie folgt konfigurieren`ECS_IMAGE_PULL_BEHAVIOR`:`prefer-cached`. Das Bild wird remote abgerufen, wenn kein zwischengespeichertes Bild vorhanden ist. Andernfalls wird das zwischengespeicherte Image in der Instance verwendet. Die automatische Bildbereinigung ist für den Container deaktiviert, um sicherzustellen, dass das zwischengespeicherte Bild nicht entfernt wird. Dadurch wird die Abrufzeit für das Abrufen von Bildern für nachfolgende Starts reduziert. Der Effekt von Caching ist noch größer, wenn Sie eine hohe Aufgabendichte in Ihren Container-Instances haben, die Sie mithilfe der `binpack` Platzierungsstrategie konfigurieren können. Das Zwischenspeichern von Container-Images ist besonders für Windows-basierte Workloads von Vorteil, die in der Regel große Container-Image-Größen (mehrere zehn GB) haben. Wenn Sie die `binpack` Platzierungsstrategie verwenden, können Sie auch erwägen, Elastic Network Interface (ENI) -Trunking zu verwenden, um mehr Aufgaben im `awsvpc` Netzwerkmodus

auf jeder Container-Instance zu platzieren. ENI-Trunking erhöht die Anzahl der Aufgaben, die Sie im Modus ausführen `awsvpc` können. Beispielsweise kann eine c5.large-Instance, die möglicherweise nur die gleichzeitige Ausführung von 2 Aufgaben unterstützt, bis zu 10 Aufgaben mit ENI-Trunking ausführen.

- Wählen Sie einen optimalen Netzwerkmodus

  Obwohl es viele Fälle gibt, in denen der `awsvpc` Netzwerkmodus ideal ist, kann dieser Netzwerkmodus von Natur aus die Latenz beim Starten von Aufgaben erhöhen, da Amazon ECS-Workflows für jede Aufgabe im `awsvpc` Modus eine ENI bereitstellen und anhängen müssen, indem sie Amazon EC2 EC2-APIs aufrufen, was Ihren Aufgabenstarts einen Mehraufwand von mehreren Sekunden hinzufügt. Im Gegensatz dazu besteht ein entscheidender Vorteil des `awsvpc` Netzwerkmodus darin, dass jede Aufgabe über eine Sicherheitsgruppe verfügt, die den Datenverkehr zulässt oder verweigert. Das bedeutet, dass Sie flexibler sind, um die Kommunikation zwischen Aufgaben und Diensten detaillierter zu steuern. Wenn die Bereitstellungsgeschwindigkeit Ihre Priorität ist, können Sie den `bridge` Modus in Betracht ziehen, um das Starten von Aufgaben zu beschleunigen. Weitere Informationen finden Sie unter the section called “AWSVPC Netzwerkmodus”.
- Verfolgen Sie Ihren Lebenszyklus beim Starten von Aufgaben, um Optimierungsmöglichkeiten zu finden

  Es ist oft schwierig zu wissen, wie lange es dauert, bis Ihre Anwendung gestartet wird. Das Starten Ihres Container-Images, das Ausführen von Startskripten und andere Konfigurationen während des Anwendungsstarts können überraschend viel Zeit in Anspruch nehmen. Sie können den Task-Metadaten-Endpunkt verwenden, um Metriken zu veröffentlichen, mit denen Sie die Startzeit der Anwendung von `ContainerStartTime` bis zu dem Zeitpunkt verfolgen können, zu dem Ihre Anwendung bereit ist, Datenverkehr bereitzustellen. Anhand dieser Daten können Sie nachvollziehen, wie Ihre Anwendung zur Gesamtstartzeit beiträgt, und Bereiche ermitteln, in denen Sie unnötigen anwendungsspezifischen Aufwand reduzieren und Ihre Container-Images optimieren können. Weitere Informationen finden Sie unter Optimieren Sie die Kapazität und Verfügbarkeit von Amazon ECS.
- Wählen Sie einen optimalen Instance-Typ (für den EC2-Starttyp)

  Die Auswahl des richtigen Instance-Typs basiert auf der Ressourcenreservierung (z. B. CPU, Arbeitsspeicher), die Sie für Ihre Aufgabe konfigurieren. Daher können Sie bei der Dimensionierung der Instanz berechnen, wie viele Aufgaben einer einzelnen Instanz zugewiesen werden können. Ein einfaches Beispiel für eine gut platzierte Aufgabe ist das Hosten von 4 Aufgaben, die 0,5 vCPU und 2 GB an Speicherreservierungen in einer m5.large-Instance erfordern (unterstützt 2 vCPUs

und 8 GB Arbeitsspeicher). Die Reservierungen dieser Aufgabendefinition nutzen die Ressourcen der Instanz voll aus.

Eine Anwendung als Amazon ECS-Aufgabe ausführen

Mit dem können Sie eine Aufgabe für einen einmaligen Vorgang erstellen AWS Management Console.

Um eine eigenständige Aufgabe zu erstellen (AWS Management Console)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Die Amazon ECS-Konsole ermöglicht es Ihnen, eine eigenständige Aufgabe entweder von Ihrer Cluster-Detailseite oder von der Revisionsliste der Aufgabendefinition aus zu erstellen. Gehen Sie je nach der ausgewählten Ressourcenseite wie folgt vor, um Ihre eigenständige Aufgabe zu erstellen.

So starten Sie einen Service von	Schritte
eine Cluster-Detailseite...	a. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie im Service erstellen möchten. b. Von der Registerkarte Tasks (Aufgaben) wählen Sie Ausführen einer neuen Aufgabe.
eine Revisionsseite für die Aufgabendefinition...	a. Wählen Sie auf der Seite mit den Aufgabendefinitionen die Aufgabendefinitionsfamilie aus, um die Revisionen für diese Familie anzuzeigen. b. Wählen Sie die Revision aus, die Sie verwenden möchten.

So starten Sie einen Service von	Schritte
	c. Wählen Sie im Menü Bereitstellen die Option Task ausführen.

3. (Optional) Im Abschnitt Compute-Konfiguration (erweitert) wählen Sie aus, wie Ihre Aufgaben verteilt werden sollen. Sie können entweder eine Capacity-Provider-Strategie oder einen Launch-Typ verwenden. Um eine Kapazitätsanbieterstrategie zu verwenden, müssen Sie Ihre Kapazitätsanbieter auf Clusterebene konfigurieren. Wenn Sie Ihren Cluster nicht für die Verwendung eines Kapazitätsanbieters konfiguriert haben, verwenden Sie stattdessen einen Starttyp.

Verteilungsmethode	Schritte
Kapazitätsanbieterstrategie	a. Wählen Sie im Bereich Compute options (Datenverarbeitungs-Optionen) die Option Capacity provider strategy (Kapazitätsanbiete rstrategie) aus. b. Wählen Sie eine Strategie aus: • Um die standardmäßige Kapazitätsanbieter-Strategie des Clusters zu verwenden, wählen Sie Use cluster default (Cluster-Standard verwenden). • Wenn Ihr Cluster keine Standardstrategie für Kapazitätsanbieter hat oder eine benutzerd

Verteilungsmethode	Schritte
	efinierte Strategie verwendet werden soll, wählen Sie Use custom (Benutzerdefiniert nutzen), Add capacity provider strategy (Kapazitätsanbieter-Strategie hinzufüge n) und definieren Sie Ihre benutzerdefinierte Kapazitätsanbieter-Strategie, indem Sie eine Basis (Base), einen Kapazitätsanbieter (Capacity provider) und ein Gewicht (Weight) angeben. **Note** Damit Kapazität sanbieter in einer Strategie verwendet werden kann, muss er dem Cluster zugeordnet sein.

Verteilungsmethode	Schritte
Starttyp	a. Wählen Sie im Bereich Compute options (Datenverarbeitungs-Optionen) die Option Launch type (Starttyp) aus. b. Wählen Sie unter Launch type (Starttyp) einen Starttyp aus. c. (Optional) Wenn der Fargate Starttyp angegeben ist, geben Sie für Plattformversion die zu verwendende Plattformversion an. Ist keine Plattformversion angegeben, wird die Plattformversion `LATEST` verwendet.

4. Für Anwendungstyp, wählen Sie Aufgabe aus.
5. Wählen Sie für Aufgabendefinition die Aufgabendefinitionsfamilie und die Version aus.

> **Important**
>
> Die Konsole validiert die Auswahl, um sicherzustellen, dass die ausgewählte Aufgabendefinitionsfamilie und -version mit der definierten Rechenkonfiguration kompatibel sind.

6. Geben Sie für Desired tasks (Gewünschte Aufgaben) die Anzahl der Aufgaben an, die gestartet werden sollen.
7. Wenn Ihre Aufgabendefinition `awsvpc`-Netzwerkmodus nutzt, erweitern Sie Networking (Netzwerk). Führen Sie die folgenden Schritte aus, um eine benutzerdefinierte Konfiguration anzugeben.

a. Wählen Sie für VPC die VPC aus, die Sie verwenden möchten.

b. Wählen Sie für Subnets (Subnetze) ein oder mehrere Subnetze in der VPC aus, die der Aufgaben-Scheduler bei der Platzierung Ihrer Aufgaben berücksichtigen soll.

> **Important**
>
> Für den Netzwerkmodus`awsvpc` werden nur private Subnetze unterstützt. Die Aufgaben erhalten keine öffentlichen IP-Adressen. Daher ist ein NAT-Gateway für ausgehenden Internet-Zugriff erforderlich und eingehender Internetdatenverkehr wird über einen Load Balancer weitergeleitet.

c. Für die Sicherheitsgruppe können Sie entweder eine vorhandene Sicherheitsgruppe auswählen oder eine neue erstellen. Um eine vorhandene Sicherheitsgruppe zu verwenden, wählen Sie die Sicherheitsgruppe aus und fahren Sie mit dem nächsten Schritt fort. Um eine neue Sicherheitsgruppe zu erstellen, wählen Sie Create a new security group. Sie müssen einen Sicherheitsgruppennamen und eine Beschreibung angeben und dann eine oder mehrere eingehende Regeln für die Sicherheitsgruppe hinzufügen.

d. Geben Sie für die Öffentliche IP an, ob der Elastic-Network-Schnittstelle (ENI) der Aufgabe eine öffentliche IP-Adresse automatisch zugewiesen wird.

   AWS Fargate Aufgaben kann eine öffentliche IP-Adresse zugewiesen werden, wenn sie in einem öffentlichen Subnetz ausgeführt werden, sodass sie eine Route zum Internet haben. Weitere Informationen finden Sie unter Fargate-Aufgabenvernetzung im Benutzerhandbuch zum Amazon Elastic Container Service für AWS Fargate.

8. Wenn Ihre Aufgabe ein Datenvolume verwendet, das mit der Konfiguration bei der Bereitstellung kompatibel ist, können Sie das Volume konfigurieren, indem Sie Volume erweitern.

   Der Datenträgername und der Volumetyp werden bei der Erstellung einer Revision der Aufgabendefinition konfiguriert und können nicht geändert werden, wenn Sie eine eigenständige Aufgabe ausführen. Um den Namen und den Typ des Volumes zu aktualisieren, müssen Sie eine neue Version der Aufgabendefinition erstellen und eine Aufgabe mithilfe der neuen Version ausführen.

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
Amazon EBS	a. Wählen Sie unter EBS-Volumetyp den Typ des EBS-Volumes aus, den Sie Ihrer Aufgabe zuordnen möchten. b. Geben Sie für Größe (GiB) einen gültigen Wert für die Datenträgergröße in Gibibyte (GiB) ein. Sie können eine Volumengröße von mindestens 1 GiB und eine maximale Volumengröße von 16.384 GiB angeben. Dieser Wert ist erforderlich, sofern Sie keine Snapshot-ID angeben. c. Geben Sie für IOPS die maximale Anzahl von Eingabe-/Ausgabevorgängen (IOPS) ein, die das Volume bereitstellen soll. Dieser Wert ist nur für die Volumetypen `io1` `io2`, und konfigurierbar. `gp3` d. Geben Sie für Durchsatz (MIB/s) den Durchsatz in Mebibyte pro Sekunde (MiBpsoder MIB/s) ein, den das Volume bereitstellen soll. Dieser Wert ist	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	nur für den Volumetyp konfigurierbar. gp3 e. Wählen Sie für Snapshot-ID einen vorhandenen Amazon EBS-Volume-Snapshot aus oder geben Sie den ARN eines Snapshots ein, wenn Sie ein Volume aus einem Snapshot erstellen möchten. Sie können auch ein neues, leeres Volume erstellen, indem Sie keine Snapshot-ID auswählen oder eingeben. f. Deaktivieren Sie das Kontrollkästchen unter Kündigungsrichtlinie, wenn Sie möchten, dass das Volume, das für das Anhängen an die Aufgabe konfiguriert ist, auch nach dem Beenden der Aufgabe erhalten bleibt. Standardmäßig werden EBS-Volumes, die an Aufgaben angehängt sind, gelöscht, wenn die Aufgabe beendet wird. g. Wählen Sie unter Dateisystemtyp den Typ des Dateisystems aus, das für das Speichern und	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Abrufen von Daten auf dem Volume verwendet werden soll. Sie können entweder den Betriebss ystemstandard oder einen bestimmten Dateisyst emtyp wählen. Die Standardeinstellung für Linux ist`XFS`. Für Volumes, die aus einem Snapshot erstellt wurden, müssen Sie denselben Dateisystemtyp angeben, den das Volume bei der Erstellung des Snapshots verwendet hat. Wenn der Dateisystemtyp nicht übereinstimmt, kann die Aufgabe nicht gestartet werden. h. Wählen Sie für die Infrastrukturrolle eine IAM-Rolle mit den erforderlichen Berechtig ungen, die es Amazon ECS ermöglichen, Amazon EBS-Volum es für Aufgaben zu verwalten. Sie können die `AmazonECSInfrastru ctureRole PolicyForVolumes` verwaltete Richtlinie an die Rolle anhängen, oder	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Sie können die Richtlinie als Leitfaden verwenden , um eine eigene Richtlini e mit Berechtigungen zu erstellen und anzuhänge n, die Ihren spezifischen Anforderungen entsprech en. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unterIAM-Rolle für die Amazon ECS-Infra struktur. i. Wählen Sie unter Verschlüsselung die Option Standard, wenn Sie die Amazon EBS-Verschlüsselung standardmäßig verwenden möchten. Wenn für Ihr Konto standardmäßig Verschlüsselung konfiguri ert ist, wird das Volume mit dem Schlüssel AWS Key Management Service (AWS KMS) verschlüs selt, der in der Einstellu ng angegeben ist. Wenn Sie Standard wählen und die Amazon EBS-Standardverschlüsselung nicht aktiviert ist, wird das Volume unverschlüsselt.	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Wenn Sie Benutzerdefiniert wählen, können Sie eine Option Ihrer Wahl für AWS KMS key die Volumenverschlüsselung angeben. Wenn Sie „Keine“ wählen, wird das Volume unverschlüsselt, es sei denn, Sie haben die Verschlüsselung standardmäßig konfiguriert oder Sie erstellen ein Volume aus einem verschlüsselten Snapshot. j. Wenn Sie Benutzerdefiniert für Verschlüsselung ausgewählt haben, müssen Sie angeben AWS KMS key , welche Sie verwenden möchten. Wählen Sie für KMS-Schlüssel einen Schlüssel-ARN aus AWS KMS key oder geben Sie einen ein. Wenn Sie Ihr Volume mithilfe eines symmetrischen, vom Kunden verwalteten Schlüssels verschlüsseln möchten, stellen Sie sicher, dass Sie in Ihrer	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	AWS KMS key Richtlinie über die richtigen Berechtigungen verfügen. Weitere Informationen finden Sie unter Datenverschlüsselung für Amazon EBS-Volumes. k. (Optional) Unter Tags können Sie Ihrem Amazon EBS-Volume Tags hinzufügen, indem Sie entweder Tags aus der Aufgabendefinition weitergeben oder Ihre eigenen Tags angeben. Wenn Sie Tags aus der Aufgabendefinition weitergeben möchten, wählen Sie Aufgabendefinition für Tags weitergeben aus. Wenn Sie Nicht weitergeben oder wenn Sie keinen Wert auswählen, werden die Tags nicht weitergegeben. Wenn Sie Ihre eigenen Tags angeben möchten, wählen Sie Tag hinzufügen und geben Sie dann den Schlüssel und den Wert	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	für jedes Tag ein, das Sie hinzufügen. Weitere Informationen zum Taggen von Amazon EBS-Volumes finden Sie unter Tagging Amazon EBS-Volumes.	

9. (Optional) Um eine andere als die standardmäßige Strategie zur Platzierung von Aufgaben zu verwenden, erweitern Sie Task Placement (Platzierung von Aufgaben) und wählen Sie aus den folgenden Optionen aus.

   Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

   - AZ Balanced Spread — Verteilen Sie Aufgaben auf Availability Zones und auf Container-Instances in der Availability Zone.
   - AZ Balanced BinPack — Verteilen Sie Aufgaben auf Availability Zones und auf Container-Instances mit dem geringsten verfügbaren Speicher.
   - BinPack— Verteilen Sie Aufgaben auf der Grundlage der geringsten verfügbaren CPU- oder Speichermenge.
   - Eine Aufgabe pro Host — Platzieren Sie maximal eine Aufgabe aus dem Service auf jeder Container-Instance.
   - Benutzerdefiniert — Definieren Sie Ihre eigene Strategie zur Aufgabenverteilung.

   Wenn Sie Custom (Benutzerdefiniert) wählen, definieren Sie den Algorithmus für das Platzieren von Aufgaben und die Regeln, die bei der Aufgabenplatzierung berücksichtigt werden.

   - Unter Strategy (Strategie), für Type (Typ) und Field (Feld), wählen Sie den Algorithmus und die Entität aus, die für den Algorithmus verwendet werden sollen.

     Sie können maximal 5 Strategien angeben.
   - Unter Einschränkung, für Typ und Ausdruck, wählen Sie die Regel und das Attribut für die Einschränkung aus.

Um beispielsweise die Einschränkung festzulegen, Aufgaben auf T2-Instances zu platzieren, geben Sie für Expression (Ausdruck) attribute:ecs.instance-type =~ t2.* ein.

Sie können maximal 10 Einschränkungen angeben.

10. (Optional) Um die in Ihrer Aufgabendefinition definierte Aufgaben-IAM-Rolle oder die Aufgabenausführungsrolle außer Kraft zu setzen, erweitern Sie Task overrides (Aufgaben-Überschreibungen) und führen Sie dann die folgenden Schritte aus:

    a. Wählen Sie unter Aufgabenrolle eine IAM-Rolle für diese Aufgabe aus. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

    Nur Rollen mit der Vertrauensstellung `ecs-tasks.amazonaws.com` werden angezeigt. Anweisungen zum manuellen Erstellen einer IAM-Rolle für Ihre Aufgaben finden Sie unter Die IAM-Rolle für Aufgaben erstellen.

    b. Wählen Sie für Aufgabenausführungsrolle eine Aufgabenausführungsrolle aus. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

11. (Optional) Um die Container-Befehle und Umgebungsvariablen außer Kraft zu setzen, erweitern Sie Container Overrides (Container-Überschreibungen) und erweitern Sie dann den Container.

    - Um einen anderen Befehl als den Befehl zur Aufgabendefinition an den Container zu senden, geben Sie unter Befehlsüberschreibung den Docker-Befehl ein.

      Weitere Informationen zum Docker-Befehl run finden Sie in der Docker-Run-Referenz im Docker-Referenzhandbuch.

    - Wählen Sie Add Environment Variable (Umgebungsvariable hinzufügen), um eine Umgebungsvariable hinzuzufügen. Geben Sie unter Key den Namen Ihrer Umgebungsvariable ein. Geben Sie für Value einen Zeichenfolgenwert für Ihren Umgebungswert ein (ohne die umgebenden doppelten Anführungszeichen (" ")).

      AWS umgibt die Zeichenketten mit doppelten Anführungszeichen (“ „) und übergibt die Zeichenfolge im folgenden Format an den Container:

      ```
      MY_ENV_VAR="This variable contains a string."
      ```

12. (Optional) Um Ihre Aufgabe leichter identifizieren zu können, erweitern Sie den Tags (Tags)-Bereich und konfigurieren Sie dann Ihre Tags.

Damit Amazon ECS automatisch alle neu gestarteten Aufgaben mit dem Clusternamen und den Task-Definition-Tags versieht, wählen Sie Turn on Amazon ECS managed tags (Mit Amazon ECS verwaltete Tags aktivieren) und anschließend Task definitions (Aufgabendefinitionen) aus.

Hinzufügen oder Entfernen eines Tag.

- [Ein Tag hinzufügen] Wählen Sie Add tag (Tag hinzufügen) und führen Sie dann das Folgende aus:
  - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
  - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
- [Tag entfernen] Wählen Sie neben dem Tag die Option Remove tag (Tag löschen) aus.

13. Wählen Sie Erstellen.

Verwenden von Amazon EventBridge Scheduler zur Planung von Amazon ECS-Aufgaben

EventBridge Scheduler ist ein serverloser Scheduler, mit dem Sie Aufgaben von einem zentralen, verwalteten Dienst aus erstellen, ausführen und verwalten können. Er bietet Funktionen zur einmaligen und wiederkehrenden Terminplanung, unabhängig von Event-Bussen und Regeln. EventBridge Scheduler ist hochgradig anpassbar und bietet eine verbesserte Skalierbarkeit im EventBridge Vergleich zu geplanten Regeln sowie ein breiteres Spektrum an API-Zieloperationen und AWS -diensten. EventBridge Scheduler bietet die folgenden Zeitpläne, die Sie für Ihre Aufgaben in der EventBridge Scheduler-Konsole konfigurieren können:

- Ratenbasiert
- Cron-basiert

  Sie können Cron-basierte Zeitpläne in jeder Zeitzone konfigurieren.
- Einmalige Zeitpläne

  Sie können Einmalpläne in jeder Zeitzone konfigurieren.

Sie können Ihr Amazon ECS mit Amazon EventBridge Scheduler planen.

Obwohl Sie eine geplante Aufgabe in der Amazon ECS-Konsole erstellen können, bietet die EventBridge Scheduler-Konsole derzeit mehr Funktionen.

Führen Sie die folgenden Schritte aus, bevor Sie eine Aufgabe planen:

1. Verwenden Sie die VPC-Konsole, um die Subnetz-IDs, in denen die Aufgaben ausgeführt werden, und die Sicherheitsgruppen-IDs für die Subnetze abzurufen. Weitere Informationen finden Sie unter Ihre Subnetze anzeigen und Ihre Sicherheitsgruppen anzeigen im Amazon-VPC-Benutzerhandbuch.
2. Konfigurieren Sie die EventBridge Scheduler-Ausführungsrolle. Weitere Informationen finden Sie unter Einrichten der Ausführungsrolle im Amazon EventBridge Scheduler-Benutzerhandbuch.

So erstellen Sie einen neuen Zeitplan mithilfe der Konsole

1. Öffnen Sie die Amazon EventBridge Scheduler-Konsole unter https://console.aws.amazon.com/scheduler/home.
2. Wählen Sie auf der Seite Zeitpläne die Option Zeitplan erstellen aus.
3. Gehen Sie auf der Seite Zeitplandetails angeben im Abschnitt Zeitplanname und -beschreibung wie folgt vor:
   a. Geben Sie unter Zeitplanname einen Namen für Ihren Zeitplan ein. z. B. **`MyTestSchedule`**.
   b. (Optional) Geben Sie unter Beschreibung eine Beschreibung für Ihren Zeitplan ein. z. B. **`TestSchedule`**.
   c. Wählen Sie für Zeitplangruppe eine Zeitplangruppe aus. Wenn Sie noch keine Gruppe haben, wählen Sie Standard. Um eine Zeitplangruppe zu erstellen, wählen Sie Eigenen Zeitplan erstellen.

      Sie verwenden Zeitplangruppen, um Tags zu Zeitplangruppen hinzuzufügen.
4. Wählen Sie Ihre Zeitplanoptionen.

Vorkommen	Vorgehensweise	
Einmaliger Zeitplan Ein einmaliger Zeitplan ruft ein Ziel nur einmal zu dem von Ihnen angegebenen Datum und der angegebenen Uhrzeit auf.	Gehen Sie für Datum und Uhrzeit wie folgt vor: • Geben Sie ein gültiges Datum im `YYYY/MM/DD` -Format ein.	

Vorkommen	Vorgehensweise	
	• Geben Sie einen Zeitstempel im 24-Stunden-Format (`hh:mm`) ein. • Wählen Sie unter Zeitzone die Zeitzone aus.	

Vorkommen	Vorgehensweise
Wiederkehrender Zeitplan Ein wiederkehrender Zeitplan ruft ein Ziel mit einer Rate auf, die Sie mit einem cron-Ausdruck oder einem Rate-Ausdruck angeben.	a. Gehen Sie bei Zeitplantyp wie folgt vor: • Um den Zeitplan mithilfe eines Cron-Ausdrucks zu definieren, wählen Sie Cron-basierter Zeitplan und geben Sie den Cron-Ausdruck ein. • Um den Zeitplan mithilfe eines Rate-Ausdrucks zu definieren, wählen Sie Rate-Ausdruck ein. Weitere Informationen zu Cron- und Rate-Ausdrücken finden Sie unter Zeitplantypen auf EventBridge Scheduler im Amazon EventBridge Scheduler-Benutzerhandbuch. b. Wählen Sie für Flexibles Zeitfenster die Option Aus, um die Option zu deaktivieren, oder wählen Sie eines der vordefinierten Zeitfenster aus. Wenn Sie beispielsweise 15 Minuten auswählen und einen wiederkehrenden Zeitplan festlegen, der sein Ziel einmal pro Stunde aufruft, wird der Zeitplan innerhalb von

Vorkommen	Vorgehensweise	
	15 Minuten nach Beginn jeder Stunde ausgeführt.	

5. (Optional) Wenn Sie im vorherigen Schritt Wiederkehrender Zeitplan ausgewählt haben, gehen Sie im Abschnitt Zeitrahmen wie folgt vor:

   a. Wählen Sie unter Zeitzone eine Zeitzone aus.

   b. Geben Sie für Startdatum und -uhrzeit ein gültiges Datum im `YYYY/MM/DD`-Format ein und geben Sie dann einen Zeitstempel im 24-Stunden-Format (`hh:mm`) an.

   c. Geben Sie für Enddatum und -uhrzeit ein gültiges Datum im `YYYY/MM/DD`-Format ein und geben Sie dann einen Zeitstempel im 24-Stunden-Format (`hh:mm`) an.

6. Wählen Sie Weiter aus.

7. Auf der Seite Ziel auswählen gehen Sie wie folgt vor:

   a. Wählen Sie Alle APIs und geben Sie dann im Suchfeld ECS ein.

   b. Wählen Sie Amazon ECS aus.

   c. Geben Sie in das Suchfeld ein und wählen Sie RunTaskdann. RunTask

   d. Wählen Sie für ECS-Cluster den Cluster aus.

   e. Wählen Sie für ECS-Aufgabe die Aufgabendefinition aus, die für die Aufgabe verwendet werden soll.

   f. Um einen Starttyp zu verwenden, erweitern Sie Rechenoptionen und wählen Sie dann Starttyp aus. Wählen Sie dann den Starttyp aus.

      Wenn der Fargate Starttyp angegeben ist, geben Sie für Plattformversion die zu verwendende Plattformversion an. Wenn keine Plattform angegeben ist, wird die Plattformversion `LATEST` verwendet.

   g. Geben Sie für Subnetze die Subnetz-IDs ein, in denen die Aufgabe ausgeführt werden soll.

   h. Geben Sie für Sicherheitsgruppen die Sicherheitsgruppen-IDs für das Subnetz ein.

   i. (Optional) Um eine andere als die standardmäßige Strategie zur Platzierung von Aufgaben zu verwenden, erweitern Sie Platzierungsbeschränkung und geben Sie dann die Einschränkungen ein.

      Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert

j. (Optional) Um Ihre Aufgaben leichter identifizieren zu können, konfigurieren Sie Ihre Tags unter Tags.

Wenn Sie möchten, dass Amazon ECS alle neu gestarteten Aufgaben automatisch mit den Aufgabendefinitions-Tags versieht, wählen Sie Amazon-ECS-verwaltete Tags aktivieren.

8. Wählen Sie Weiter aus.
9. Führen Sie auf der Seite Settings (Einstellungen) die folgenden Schritte aus:

a. Um den Zeitplan zu aktivieren, schalten Sie unter Zeitplanstatus die Option Zeitplan aktivieren ein.

b. Um eine Wiederholungsrichtlinie für Ihren Zeitplan zu konfigurieren, gehen Sie unter Wiederholungsrichtlinie und Warteschlange für unzustellbare Nachrichten (DLQ) wie folgt vor:

- Aktivieren Sie die Option Wiederholen.
- Geben Sie unter Maximale Aufbewahrungszeit des Ereignisses die maximale (n) Stunde (n) und Minute (n) ein, die der EventBridge Scheduler für ein unbearbeitetes Ereignis speichern muss.
- Die Höchstdauer beträgt 24 Stunden.
- Geben Sie unter Maximale Anzahl an Wiederholungen ein, wie oft der EventBridge Scheduler den Zeitplan maximal wiederholt, falls das Ziel einen Fehler zurückgibt.

  Der Maximalwert beträgt 185 Wiederholungen.

Bei Wiederholungsrichtlinien führt der Scheduler den Zeitplan erneut aus, wenn ein Zeitplan sein Ziel nicht aufrufen kann. EventBridge Falls konfiguriert, müssen Sie die maximale Aufbewahrungszeit und Wiederholungsversuche für den Zeitplan festlegen.

c. Wählen Sie aus, wo der EventBridge Scheduler nicht zugestellte Ereignisse speichert.

Option für Warteschlange für unzustellbare Nachrichten (DLQ)	Vorgehensweise	
Nicht speichern	Wählen Sie None.	

Option für Warteschlange für unzustellbare Nachrichten (DLQ)	Vorgehensweise	
Speichern Sie das Ereignis dort, AWS-Konto wo Sie den Zeitplan erstellen	a. Wählen Sie Wählen Sie eine Amazon SQS SQS-Warteschlange in meiner AWS-Konto als DLQ aus. b. Wählen Sie den Amazon-Ressourcennamen (ARN) der Amazon SQS-Warteschlange.	
Speichern Sie das Ereignis an einem anderen Ort als AWS-Konto dem, an dem Sie den Zeitplan erstellen	a. Wählen Sie Eine Amazon SQS SQS-Warteschlange in einer anderen AWS-Konten als DLQ angeben aus. b. Geben Sie den Amazon-Ressourcennamen (ARN) der Amazon-SQS-Warteschlange ein.	

d. Um einen kundenverwalteten Schlüssel zur Verschlüsselung Ihrer Zieleingabe zu verwenden, wählen Sie unter Verschlüsselung die Option Verschlüsselungseinstellungen anpassen (erweitert).

   Wenn Sie diese Option wählen, geben Sie einen vorhandenen CMK-ARN ein oder wählen Sie Erstellen eines AWS KMS key, um zur AWS KMS -Konsole zu navigieren. Weitere Informationen darüber, wie EventBridge Scheduler Ihre Daten im Ruhezustand verschlüsselt, finden Sie unter Verschlüsselung im Ruhezustand im Amazon EventBridge Scheduler-Benutzerhandbuch.

e. Wählen Sie für Berechtigungen die Option Bestehende Rolle verwenden und wählen Sie dann die Rolle aus.

Damit EventBridge Scheduler eine neue Ausführungsrolle für Sie erstellt, wählen Sie Neue Rolle für diesen Zeitplan erstellen. Geben Sie dann einen Namen für Rollenname ein. Wenn Sie diese Option wählen, ordnet EventBridge Scheduler der Rolle die erforderlichen Berechtigungen zu, die für Ihr vorgegebenes Ziel erforderlich sind.

10. Wählen Sie Weiter aus.
11. Überprüfen Sie auf der Seite Zeitplan überprüfen und erstellen die Details Ihres Zeitplans. Wählen Sie in jedem Abschnitt Bearbeiten aus, um zu diesem Schritt zurückzukehren und seine Details zu bearbeiten.
12. Wählen Sie Zeitplan erstellen.

    Auf der Seite Zeitpläne können Sie eine Liste Ihrer neuen und vorhandenen Zeitpläne anzeigen. Überprüfen Sie in der Spalte Status, ob Ihr neuer Zeitplan aktiviert ist.

Nächste Schritte

Sie können die EventBridge Scheduler-Konsole oder die verwenden, AWS CLI um den Zeitplan zu verwalten. Weitere Informationen finden Sie unter Einen Zeitplan verwalten im Amazon EventBridge Scheduler-Benutzerhandbuch.

Eine Amazon ECS-Aufgabe beenden

Wenn Sie eine eigenständige Aufgabe nicht mehr ausführen müssen, können Sie die Aufgabe beenden. Die Amazon ECS-Konsole macht es einfach, eine oder mehrere Aufgaben zu beenden.

Wenn Sie einen Service beenden möchten, finden Sie weitere Informationen unterLöschen eines Amazon ECS-Service mithilfe der Konsole.

Um eine eigenständige Aufgabe zu beenden (AWS Management Console)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Seite „Cluster“ den Cluster aus, um zur Seite mit den Cluster-Details zu navigieren.
4. Wählen Sie auf der Cluster-Detailseite die Registerkarte Aufgaben aus.
5. Mithilfe der Liste Starttyp filtern können Sie Aufgaben nach Starttyp filtern.

Zu stoppende Aufgaben	Schritte
Eine oder mehrere	a. Wählen Sie die Aufgaben aus und klicken Sie dann auf Beenden, Ausgewählte beenden. b. Wählen Sie auf der Bestätigungsseite „Aufgabe beenden“ die Option Beenden

Zu stoppende Aufgaben	Schritte
Alle	**Important** Wenn Sie sich dafür entscheiden, alle Aufgaben mithilfe der Konsole zu beenden, stoppt Amazon ECS alle eigenstän digen Aufgaben und Aufgaben, die Teil eines Service sind. Daher raten wir zur Vorsicht bei der Verwendung dieser Option. a. Wählen Sie Stopp, Alle stoppen. b. Geben Sie auf der Bestätigungsseite „Aufgabe beenden“ den Text Alle Aufgaben beenden ein, und wählen Sie dann Beenden aus.

Amazon-ECS-Dienstleistungen

Sie können einen Amazon-ECS-Service verwenden, um eine bestimmte Anzahl von Instances einer Aufgabendefinition gleichzeitig in einem Amazon-ECS-Cluster auszuführen und zu verwalten. Wenn eine Ihrer Aufgaben ausfällt oder anhält, startet der Amazon-ECS-Service-Scheduler eine andere Instance Ihrer Aufgabendefinition, um sie zu ersetzen. Dies trägt dazu bei, dass die von Ihnen gewünschte Anzahl von Aufgaben im Service erhalten bleibt.

Sie können Ihren Service optional auch hinter einem Load Balancer ausführen. Der Load Balancer verteilt den Datenverkehr über die mit dem Service verbundenen Aufgaben.

Wir empfehlen die Verwendung des Service-Schedulers für lang laufende zustandslose Services und Anwendungen. Der Service Scheduler stellt sicher, dass die von Ihnen angegebene Einplanungsstrategie einhalten wird, und plant Aufgaben neu ein, wenn eine Aufgabe fehlschlägt. Wenn beispielsweise die zugrunde liegende Infrastruktur ausfällt, plant der Service-Scheduler eine Aufgabe neu ein. Sie können Strategien zur Aufgabenplatzierung und Einschränkungen verwenden, um die Platzierung und Beendigung von Aufgaben durch den Scheduler anzupassen. Wenn eine Aufgabe in einem Service gestoppt wird, startet der Scheduler eine neue Aufgabe, die diese ersetzt. Dieser Vorgang wird so lange fortgesetzt, bis Ihr Service die gewünschte Anzahl von Aufgaben erreicht hat, basierend auf der Planungsstrategie, die der Service verwendet. Die Planungsstrategie des Services wird auch als Servicetyp bezeichnet.

Der Service-Scheduler ersetzt auch Aufgaben, die nach einem Fehlschlagen einer Container-Zustandsprüfung oder einer Load-Balancer-Zielgruppen-Zustandsprüfung als fehlerhaft eingestuft wurden. Dieser Ersatz hängt von den Parametern `maximumPercent` und `desiredCount` der Servicedefinition ab. Wenn eine Aufgabe als fehlerhaft markiert ist, startet der Service-Scheduler zunächst eine Ersatzaufgabe. Dann passiert Folgendes.

- Wenn die Ersatzaufgabe den Integritätsstatus hat`HEALTHY`, stoppt der Service Scheduler die fehlerhafte Aufgabe
- Wenn die Ersatzaufgabe den Zustand `UNHEALTHY` hat, stoppt der Scheduler entweder die fehlerhafte Ersatzaufgabe oder die vorhandene fehlerhafte Aufgabe, sodass die Gesamtanzahl der Aufgaben auf einen Wert gleich `desiredCount` eingestellt wird.

Wenn der Parameter `maximumPercent` den Scheduler daran hindert, zuerst eine Ersatzaufgabe zu starten, stoppt der Scheduler fehlerhafte Aufgaben einzeln nach dem Zufallsprinzip, um Kapazität freizugeben, und startet dann eine Ersatzaufgabe. Der Start- und Stoppvorgang wird fortgesetzt, bis alle fehlerhaften Aufgaben durch fehlerfreie Aufgaben ersetzt wurden. Sobald alle fehlerhaften Aufgaben ersetzt wurden und nur noch fehlerfreie Aufgaben ausgeführt werden, werden, wenn die Gesamtzahl der Aufgaben `desiredCount` übersteigt, die fehlerfreien Aufgaben nach dem Zufallsprinzip angehalten, bis die Gesamtzahl der Aufgaben gleich `desiredCount` ist. Weitere Informationen zu `maximumPercent` und `desiredCount` finden Sie unter Servicedefinitionsparamater.

Der Service-Scheduler enthält eine Logik, die die Häufigkeit des Neustarts von Aufgaben drosselt, wenn Aufgaben wiederholt nicht gestartet werden. Wenn eine Aufgabe gestoppt wird, ohne dass

sie in einen RUNNING-Status eingetreten ist, beginnt der Service-Scheduler, die Startversuche zu verlangsamen, und sendet eine Service-Ereignismeldung aus. Dieses Verhalten verhindert, dass unnötige Ressourcen für fehlgeschlagene Aufgaben verwendet werden, bevor Sie das Problem beheben können. Nachdem der Service aktualisiert wurde, nimmt der Service-Scheduler sein normales Planungsverhalten wieder auf. Weitere Informationen finden Sie unter Drosselungslogik für Amazon ECS-Services und Ereignismeldungen des Amazon ECS-Service anzeigen.

Es gibt zwei Strategien für Service-Scheduler:

- REPLICA: Die Replica-Einplanungsstrategie platziert und die gewünschte Anzahl von Aufgaben in Ihrem Cluster und behält sie bei. Standardmäßig verteilt der Service-Scheduler Aufgaben über Availability Zones. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie festlegen, wie Aufgaben platziert und beendet werden. Weitere Informationen finden Sie unter Replikat-Strategie.
- DAEMON: Die Daemon-Einplanungsstrategie stellt genau eine Aufgabe auf jeder aktiven Container-Instance bereit, die alle von Ihnen in Ihrem Cluster angegebenen Platzierungsbedingungen für die Aufgaben erfüllt. Bei Verwendung dieser Strategie ist es nicht erforderlich, eine gewünschte Anzahl von Aufgaben oder eine Aufgabenplatzierungsstrategie anzugeben oder Auto-Scaling-Richtlinien zu verwenden. Weitere Informationen finden Sie unter Daemon-Strategie.

> **Note**
>
> Fargate-Aufgaben unterstützen die DAEMON-Einplanungsstrategie nicht.

Daemon-Strategie

Die Daemon-Planungsstrategie stellt genau eine Aufgabe auf jeder aktiven Container-Instance bereit, die alle Platzierungseinschränkungen für die Aufgaben in Ihrem Cluster erfüllt. Der Service Scheduler bewertet die Einschränkungen bei der Aufgabenplatzierung für laufende Aufgaben und beendet Aufgaben, die die Platzierungsbeschränkungen nicht erfüllen. Wenn Sie diese Strategie verwenden, müssen Sie weder die gewünschte Anzahl von Aufgaben noch eine Strategie zur Aufgabenplatzierung angeben oder Service Auto Scaling Scaling-Richtlinien verwenden.

Amazon ECS reserviert Container-Instance-Computing-Ressourcen, einschließlich CPU, Arbeitsspeicher und Netzwerkschnittstellen für die Daemon-Aufgaben. Wenn Sie einen Daemon-Service in einem Cluster mit anderen Replikat-Services starten, priorisiert Amazon ECS die Daemon-Aufgabe. Das bedeutet, dass die Daemon-Aufgabe die erste Aufgabe ist, die auf den Instances gestartet wird, und die letzte Aufgabe, die beendet wird, nachdem alle Replikataufgaben

beendet wurden. Diese Strategie stellt sicher, dass die Ressourcen nicht von anhängigen Replikationsaufgaben verwendet werden und für die Daemon-Aufgaben verfügbar sind.

Der Daemon-Service-Scheduler platziert keine Aufgaben auf Instances mit dem Status `DRAINING`. Wenn eine Container-Instance in einen `DRAINING`-Status übergeht, werden die darauf befindlichen Daemon-Aufgaben gestoppt. Der Service Scheduler überwacht auch, ob Ihrem Cluster neue Container-Instances hinzugefügt werden, und fügt diesen die Daemon-Aufgaben hinzu.

Wenn Sie eine Bereitstellungskonfiguration angeben, muss der Wert für den `maximumPercent` Parameter `100` (als Prozentsatz angegeben) sein. Dies ist der Standardwert, der verwendet wird, wenn er nicht festgelegt ist. Der Standardwert für den `minimumHealthyPercent` Parameter ist `0` (als Prozentsatz angegeben).

Sie müssen den Service neu starten, wenn Sie die Platzierungseinschränkungen für den Daemon-Service ändern. Amazon ECS aktualisiert dynamisch die Ressourcen, die auf qualifizierten Instances für die Daemon-Aufgabe reserviert sind. Bei vorhandenen Instances versucht der Scheduler, die Aufgabe auf der Instance zu platzieren.

Eine neue Bereitstellung beginnt, wenn die Aufgabengröße oder die Reservierung von Containerressourcen in der Aufgabendefinition geändert wird. Amazon ECS nimmt die aktualisierten CPU- und Speicherreservierungen für den Daemon auf und sperrt diese Kapazität für die Daemon-Aufgabe.

Wenn für einen der oben genannten Fälle nicht genügend Ressourcen vorhanden sind, geschieht Folgendes:

- Die Aufgabenplatzierung schlägt fehl.
- Ein CloudWatch Ereignis wird generiert.
- Amazon ECS versucht weiterhin, die Aufgabe für die Instance zu planen, indem er darauf wartet, dass Ressourcen verfügbar sind.
- Amazon ECS gibt alle reservierten Instances frei, die die Platzierungseinschränkungskriterien nicht mehr erfüllen, und stoppt die entsprechenden Daemon-Aufgaben.

Die Daemon-Scheduling-Strategie kann in folgenden Fällen verwendet werden:

- Ausführen von Anwendungscontainern
- Ausführen von Support-Containern für Protokollierungs-, Überwachungs- und Tracing-Aufgaben

Aufgaben, die den Starttyp Fargate oder die Bereitstellungs-Controller-Typen `CODE_DEPLOY` oder `EXTERNAL` verwenden, unterstützen die Daemon-Scheduling-Strategie nicht.

Wenn der Service-Scheduler Aufgaben stoppt, versucht er, eine Ausgewogenheit in den Availability Zones in Ihrem Cluster herzustellen. Der Scheduler verwendet die folgende Logik:

- Wenn eine Platzierungsstrategie definiert ist, wählen Sie mit dieser Strategie die zu beendenden Aufgaben aus. Wenn beispielsweise für einen Service eine Strategie für die Verteilung der Availability Zone definiert wurde, wird eine Aufgabe ausgewählt, die die verbleibenden Aufgaben mit der besten Verteilung belässt.
- Wenn keine Platzierungsstrategie definiert ist, verwenden Sie die folgende Logik, um das Gleichgewicht zwischen den Availability Zones in Ihrem Cluster aufrechtzuerhalten:
  - Sortiert die gültigen Container-Instances. Geben Sie den Instances den Vorrang, die in ihrer jeweiligen Availability Zone die größte Anzahl an laufenden Aufgaben für diesen Service haben. Wenn beispielsweise in Zone A eine Serviceaufgabe läuft und in den Zonen B und C jeweils zwei Serviceaufgaben laufen, werden Container Instances in Zone B oder C als optimal für die Terminierung angesehen.
  - Stoppen Sie die Aufgabe auf einer Container-Instance in einer optimalen Availability Zone, basierend auf den vorherigen Schritten. Bevorzugung von Container-Instances mit der größten Anzahl von ausgeführten Aufgaben für diesen Service.

Replikat-Strategie

Die Planungsstrategie replica platziert und bewahrt die gewünschte Anzahl von Aufgaben in Ihrem Cluster.

Für einen Service, der Aufgaben auf Fargate ausführt, verwendet der Service-Scheduler, wenn er neue Aufgaben startet oder laufende Aufgaben stoppt, den bestmöglichen Ansatz, um ein Gleichgewicht zwischen den Availability Zones herzustellen. Sie müssen keine Aufgabenplatzierungsstrategien oder Beschränkungen angeben.

Wenn Sie einen Service erstellen, der Aufgaben auf EC2-Instances ausführt, können Sie optional Strategien und Einschränkungen für die Aufgabenplatzierung angeben, um die Entscheidungen zur Aufgabenplatzierung anzupassen. Wenn keine Strategien oder Einschränkungen für die Aufgabenplatzierung angegeben werden, verteilt der Service Scheduler die Aufgaben standardmäßig über Availability Zones. Der Scheduler verwendet die folgende Logik:

- Bestimmt, welche der Container-Instances in Ihrem Cluster die Aufgabendefinition Ihres Services unterstützen können (z. B. erforderliche CPU, Arbeitsspeicher, Ports und Container-Instance-Attribute).
- Bestimmt, welche Container-Instances alle Platzierungseinschränkungen erfüllen, die für den Service definiert sind.
- Wenn Sie einen Replikat-Service haben, der von einem Daemon-Service abhängt (z. B. eine Daemon-Log-Router-Aufgabe, die ausgeführt werden muss, bevor Aufgaben die Protokollierung nutzen können), erstellen Sie eine Einschränkung für die Aufgabenplatzierung, die sicherstellt, dass die Daemon-Service-Aufgaben vor den Replikat-Service-Aufgaben auf der EC2-Instance platziert werden. Weitere Informationen finden Sie unter Beispiel für Einschränkungen bei der Aufgabenplatzierung in Amazon ECS.
- Wenn es eine definierte Platzierungsstrategie gibt, verwenden Sie diese Strategie, um eine Instance aus den verbleibenden Kandidaten auszuwählen.
- Wenn es keine definierte Platzierungsstrategie gibt, verwenden Sie die folgende Logik, um Tasks auf die Availability Zones in Ihrem Cluster zu verteilen:
  - Sortiert die gültigen Container-Instances. Gibt den Instances Vorrang, die in ihrer jeweiligen Availability Zone die geringste Anzahl von laufenden Aufgaben für diesen Service haben. Wenn beispielsweise Zone A über eine ausgeführte Serviceaufgabe verfügt und die Zonen B und C jeweils über keine, werden gültige Container-Instances in Zone B oder C als optimal für eine Platzierung erachtet.
  - Platziert die neue Serviceaufgabe auf einer gültigen Container-Instance in einer optimalen Availability Zone, basierend auf den vorherigen Schritten. Begünstigt Container-Instances mit der geringsten Anzahl von ausgeführten Aufgaben für diesen Service.

Bewährte Methoden für Amazon ECS-Serviceparameter

Um sicherzustellen, dass es zu keinen Ausfallzeiten der Anwendung kommt, läuft der Bereitstellungsprozess wie folgt ab:

1. Starten Sie die neuen Anwendungscontainer, während die vorhandenen Container weiterlaufen.
2. Überprüfen Sie, ob die neuen Container fehlerfrei sind.
3. Stoppen Sie die alten Container.

Abhängig von Ihrer Bereitstellungskonfiguration und der Menge an freiem, nicht reserviertem Speicherplatz in Ihrem Cluster kann es mehrere Runden dauern, bis dieser Vorgang abgeschlossen ist und alle alten Aufgaben durch neue Aufgaben ersetzt sind.

Es gibt zwei Konfigurationsoptionen für den ECS-Service, mit denen Sie die Anzahl ändern können:

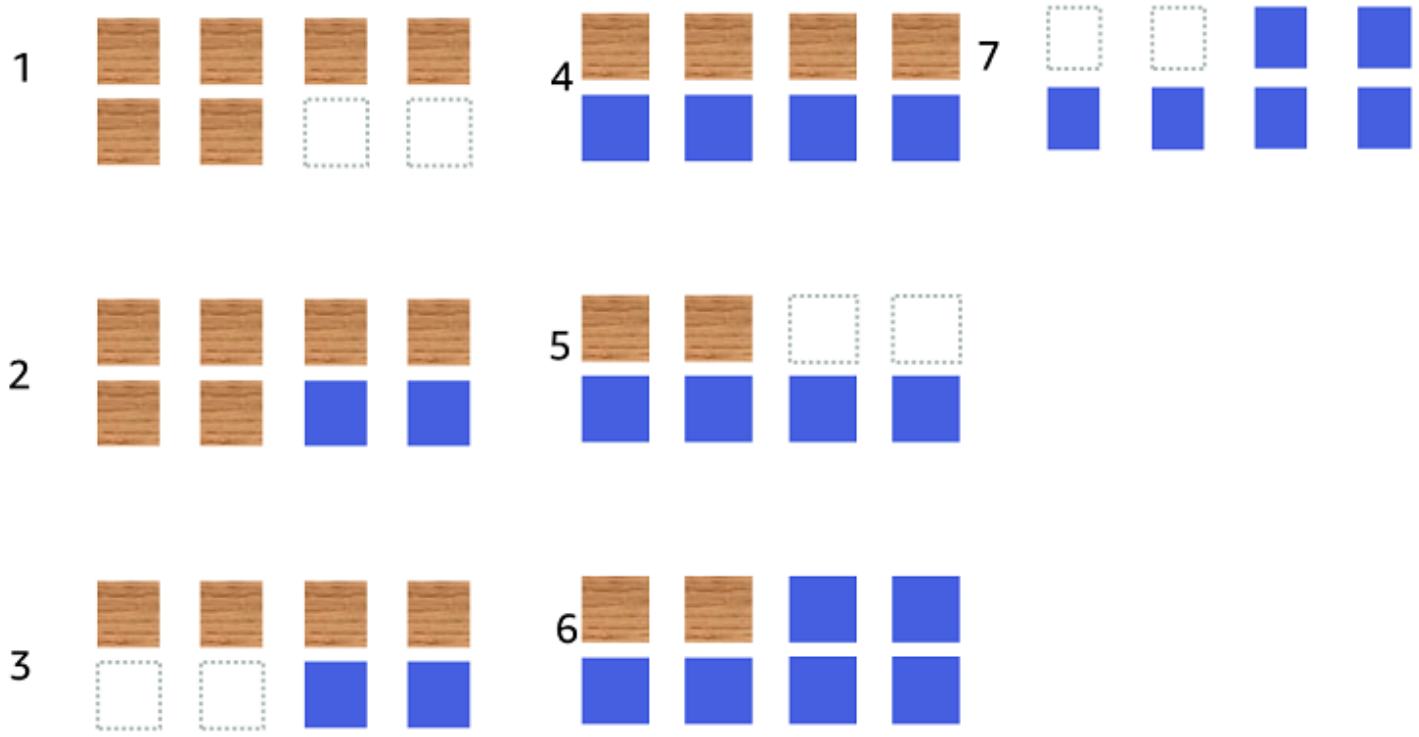
- `minimumHealthyPercent`: 100% (Standard)

  Die Untergrenze für die Anzahl der Aufgaben für Ihren Service, die während einer Bereitstellung den `RUNNING` Status beibehalten müssen. Dies ist ein auf die nächste Ganzzahl `desiredCount` aufgerundeter Prozentsatz. Mit diesem Parameter können Sie Bereitstellungen durchführen, ohne zusätzliche Clusterkapazität zu verwenden.

- `maximumPercent`: 200% (Standard)

  Die Obergrenze für die Anzahl der Aufgaben für Ihren Service, die während einer Bereitstellung im `PENDING` Status `RUNNING` oder zulässig sind. Dies ist ein Prozentsatz von, der auf die nächste Ganzzahl `desiredCount` abgerundet wurde.

Stellen Sie sich den folgenden Dienst mit sechs Tan-Aufgaben vor, die in einem Cluster bereitgestellt werden, der Platz für insgesamt acht Aufgaben bietet. Die standardmäßigen Amazon ECS-Servicekonfigurationsoptionen lassen nicht zu, dass die Bereitstellung unter 100% der sechs gewünschten Aufgaben fällt.



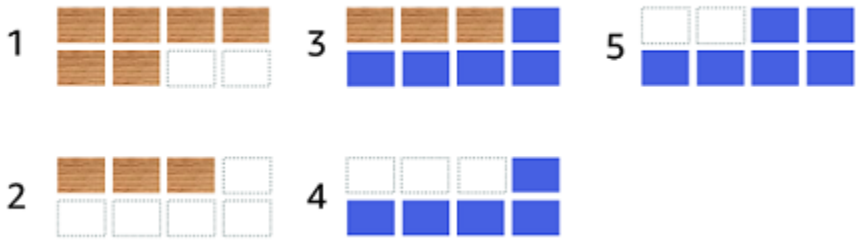
Der Bereitstellungsprozess sieht wie folgt aus:

1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.
2. Der Scheduler startet zwei neue blaue Aufgaben, da die Standardeinstellungen voraussetzen, dass sechs Aufgaben ausgeführt werden.
3. Der Scheduler stoppt zwei der hellbraunen Aufgaben, da es insgesamt sechs Aufgaben geben wird (vier hellbraune und zwei blaue).
4. Der Scheduler startet zwei zusätzliche blaue Aufgaben.
5. Der Scheduler beendet zwei der Tan-Aufgaben.
6. Der Scheduler startet zwei zusätzliche blaue Aufgaben.
7. Der Scheduler beendet die letzten beiden hellbraunen Aufgaben.

Wenn Sie im obigen Beispiel die Standardwerte für die Optionen verwenden, gibt es eine Wartezeit von 2,5 Minuten für jede neue Aufgabe, die gestartet wird. Darüber hinaus muss der Load Balancer möglicherweise 5 Minuten warten, bis die alte Aufgabe beendet ist.

Sie können die Bereitstellung beschleunigen, indem Sie den `minimumHealthyPercent` Wert auf 50% setzen.

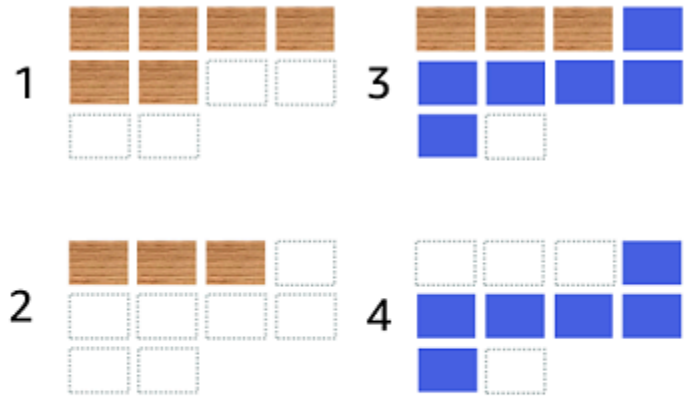
Stellen Sie sich den folgenden Dienst mit sechs Tan-Aufgaben vor, die in einem Cluster bereitgestellt werden, der Platz für insgesamt acht Aufgaben bietet.



Der Bereitstellungsprozess sieht wie folgt aus:

1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.
2. Der Scheduler stoppt drei der TAN-Aufgaben. Es werden immer noch drei TAN-Aufgaben ausgeführt, die dem `minimumHealthyPercent` Wert entsprechen.
3. Der Scheduler startet fünf blaue Aufgaben.
4. Der Scheduler stoppt die verbleibenden drei hellbraunen Aufgaben.
5. Der Scheduler startet die letzten blauen Aufgaben.

Sie können auch zusätzlichen freien Speicherplatz hinzufügen, sodass Sie zusätzliche Aufgaben ausführen können.



Der Bereitstellungsprozess sieht wie folgt aus:

1. Ziel ist es, die hellbraunen Aufgaben durch die blauen Aufgaben zu ersetzen.
2. Der Scheduler stoppt drei der TAN-Aufgaben
3. Der Scheduler startet sechs blaue Aufgaben
4. Der Scheduler stoppt die drei hellbraunen Aufgaben.

Verwenden Sie die folgenden Werte für die Amazon ECS-Servicekonfigurationsoptionen, wenn Ihre Aufgaben für einige Zeit inaktiv sind und keine hohe Nutzungsrate aufweisen.

- `minimumHealthyPercent`: 50%
- `maximumPercent`: 20%

Einen Amazon ECS-Service mithilfe der Konsole erstellen

Sie können einen Service mit der Konsole erstellen.

Die folgenden Hinweise gelten für die Verwendung der Konsole:

- Es gibt zwei Rechenoptionen, die Ihre Aufgaben verteilen.

  - Eine capacity provider strategy (Strategie für Kapazitätsanbieter) veranlasst Amazon ECS, Ihre Aufgaben an einen oder mehrere Kapazitätsanbieter zu verteilen.
  - Ein Starttyp veranlasst Amazon ECS, Ihre Aufgaben entweder direkt auf Fargate oder auf den Amazon-EC2-Instances zu starten, die für Ihre Cluster registriert sind.
- Aufgabendefinitionen, die den `awsvpc`-Netzwerkmodus oder Dienste, die für die Verwendung eines Load Balancer konfiguriert sind, verwenden, müssen über eine Netzwerkkonfiguration verfügen. Standardmäßig wählt die Konsole die standardmäßige Amazon VPC zusammen mit allen Subnetzen und der Standardsicherheitsgruppe innerhalb der standardmäßigen Amazon VPC aus.
- Die Standardstrategie zur Aufgabenplatzierung verteilt Aufgaben gleichmäßig auf die Availability Zones.
- Wenn Sie den Launch Type (Starttyp) für Ihre Servicebereitstellung verwenden, startet der Service standardmäßig in den Subnetzen in Ihrer Cluster-VPC.
- Für die Strategie für Kapazitätsanbieter, wählt die Konsole standardmäßig eine Rechenoption aus. Im Folgenden wird die Reihenfolge beschrieben, in der die Konsole einen Standardwert auswählt:
  - Wenn der Cluster eine standardmäßige Kapazitätsanbieterstrategie definiert hat, ist diese ausgewählt.
  - Wenn für Ihren Cluster keine standardmäßige Kapazitätsanbieterstrategie definiert ist, Sie aber die Fargate-Kapazitätsanbieter zum Cluster hinzugefügt haben, wird eine benutzerdefinierte Kapazitätsanbieterstrategie ausgewählt, die den `FARGATE` Kapazitätsanbieter verwendet.
  - Wenn für Ihren Cluster keine Standardstrategie für Kapazitätsanbieter definiert ist, Sie dem Cluster jedoch einen oder mehrere Auto Scaling Scaling-Gruppenkapazitätsanbieter hinzugefügt haben, ist die Option Benutzerdefiniert (erweitert) verwenden ausgewählt und Sie müssen die Strategie manuell definieren.
  - Wenn in Ihrem Cluster keine Standardstrategie für Kapazitätsanbieter definiert ist und keine Kapazitätsanbieter zum Cluster hinzugefügt wurden, ist der Fargate Starttyp ausgewählt.
- Bei den Standardoptionen für die Erkennung von Bereitstellungsfehlern wird die Option Amazon ECS Deployment Circuit Breaker zusammen mit der Option Rollback bei Fehlern verwendet.

  Weitere Informationen finden Sie unter So erkennt der Amazon ECS Deployment Circuit Breaker Fehler.
- Wenn Sie die Bereitstellungsoption Blau/Grün verwenden möchten, legen Sie fest, wie die Anwendungen CodeDeploy verschoben werden. Verfügbar sind die nachfolgend aufgeführten Optionen:

  - CodeDeployDefault.ecs AllAt Once: Verschiebt den gesamten Datenverkehr auf einmal in den aktualisierten Amazon ECS-Container
  - CodeDeploydefault.ecsLinear10 PercentEvery 1Minutes: Verschiebt jede Minute 10 Prozent des Datenverkehrs, bis der gesamte Verkehr verlagert ist.
  - CodeDeploydefault.ecsLinear10 PercentEvery 3Minutes: Verschiebt 10 Prozent des Datenverkehrs alle 3 Minuten, bis der gesamte Verkehr verlagert ist.
  - CodeDeploydefault.ecscanary10Percent5Minutes: Verschiebt 10 Prozent des Datenverkehrs in der ersten Stufe. Die restlichen 90 Prozent werden fünf Minuten später bereitgestellt.
  - CodeDeploydefault.ecscanary10Percent15Minutes: Verschiebt 10 Prozent des Datenverkehrs in der ersten Stufe. Die restlichen 90 Prozent werden 15 Minuten später bereitgestellt.
- Wenn Sie eine Anwendung mit anderen Anwendungen, die in Amazon ECS laufen, verbinden müssen, bestimmen Sie die Option, die zu Ihrer Architektur passt. Weitere Informationen finden Sie unter Amazon ECS-Services verbinden.
- Sie müssen AWS CloudFormation or verwenden, um einen Dienst bereitzustellen AWS Command Line Interface , der einen der folgenden Parameter verwendet:
  - Tracking-Richtlinie mit einer benutzerdefinierten Metrik
  - Aktualisierungsdienst — Sie können die `awsvpc` Netzwerkkonfiguration und den Kulanzzeitraum für die Integritätsprüfung nicht aktualisieren.

  Informationen zum Erstellen eines Dienstes mit dem AWS CLI finden Sie create-servicein der AWS Command Line Interface Referenz.

  Informationen zum Erstellen eines Dienstes mit AWS CloudFormation finden Sie AWS::ECS::Serviceim AWS CloudFormation Benutzerhandbuch unter.

Erstellen Sie schnell einen Service

Mit der Konsole können Sie schnell einen Service erstellen und bereitstellen. Der Service hat die folgenden Konfigurationseinstellungen:

- Wird in der VPC und in Subnetzen bereitgestellt, die Ihrem Cluster zugeordnet sind
- Stellt eine Aufgabe bereit
- Nutzt die laufende Bereitstellung
- Verwendet die Kapazitätsanbieter-Strategie mit Ihrem Standardkapazitätsanbieter

- Verwendet den Bereitstellungsschutzschalter, um Fehler zu erkennen, und legt die Option darauf fest, dass die Bereitstellung bei einem Fehler automatisch zurückgesetzt wird

Folgen Sie diesen Schritten, um einen Service mit den Standardparametern bereitzustellen.

So erstellen Sie einen Service (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Seite Cluster den Cluster aus, in dem der Service erstellt werden soll.
4. Wählen Sie auf der Registerkarte Services die Option Create (Erstellen) aus.
5. Unter Deployment configuration (Konfiguration der Bereitstellung), geben Sie an, wie Ihre Anwendung bereitgestellt wird.

   a. Für Application type (Anwendungstyp) wählen Sie Service (Service) aus.

   b. Wählen Sie für Aufgabendefinition die Aufgabendefinitionsfamilie und die zu verwendende Version aus.

   c. Wählen Sie für Service name (Servicename) einen Namen für Ihren Service aus.

   d. Geben Sie für Desired tasks (Gewünschte Aufgaben) die Anzahl der Aufgaben an, die im Service gestartet und aufrecht erhalten werden sollen.
6. (Optional) Um Ihren Service und Ihre Aufgaben besser identifizieren zu können, erweitern Sie den Bereich Tags (Tags) und konfigurieren Sie dann Ihre Tags.

   Damit Amazon ECS automatisch alle neu gestarteten Aufgaben mit dem Clusternamen und den Task-Definition-Tags versieht, wählen Sie Turn on Amazon ECS managed tags (Mit Amazon ECS verwaltete Tags aktivieren) und anschließend Task definitions (Aufgabendefinitionen) aus.

   Damit Amazon ECS automatisch alle neu gestarteten Aufgaben mit dem Clusternamen und den Service-Tags versieht, wählen Sie Turn on Amazon ECS managed tags (Mit Amazon ECS verwaltete Tags aktivieren) und anschließend Service aus.

   Hinzufügen oder Entfernen eines Tag.

   - [Ein Tag hinzufügen] Wählen Sie Add tag (Tag hinzufügen) und führen Sie dann das Folgende aus:
     - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.

- Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
- [Tag entfernen] Wählen Sie neben dem Tag die Option Remove tag (Tag löschen) aus.

Erstellen eines Services mit definierten Parametern

Gehen Sie folgendermaßen vor, um einen Service mithilfe definierter Parameter zu erstellen.

So erstellen Sie einen Service (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Bestimmen Sie die Ressource, von der aus Sie den Service starten.

So starten Sie einen Service von	Schritte
Cluster	a. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie im Service erstellen möchten. b. Wählen Sie auf der Registerkarte Services die Option Create (Erstellen) aus.
Starttyp	a. Wählen Sie auf der Seite mit den Aufgabendefinitionen das Optionsfeld neben der Aufgabendefinition aus. b. Wählen Sie im Menü Bereitstellen die Option Dienst erstellen aus.

3. (Optional) Wählen Sie aus, wie Ihre geplante Aufgaben auf Ihre Cluster-Infrastruktur verteilt werden. Erweitern Sie Datenverarbeitungskonfiguration, und wählen Sie dann Ihre Option.

Verteilungsmethode	Schritte	
Kapazitätsanbieterstrategie	a. Wählen Sie unter Rechenoptionen die Option Kapazitätsanbieter strategie aus. b. Wählen Sie eine Strategie aus: • Um die standardmäßige Kapazitätsanbieter-Strategie des Clusters zu verwenden, wählen Sie Use cluster default (Cluster-Standard verwenden). • Wenn Ihr Cluster nicht über eine Standard-Kapazitätsanbieter-Strategie verfügt oder Sie eine benutzerd efinierte Strategie verwenden möchten, wählen Sie Benutzerd efiniert verwenden, Kapazitätsanbieter-Strategie hinzufügen und definieren dann Ihre benutzerdefinierte Kapazitätsanbieter-Strategie, indem Sie eine Basis, einen Kapazitätsanbieter und ein Gewicht angeben.	

Verteilungsmethode	Schritte
	Note Damit Kapazität sanbieter in einer Strategie verwendet werden kann, muss er dem Cluster zugeordnet sein.
Starttyp	a. Wählen Sie im Bereich Compute options (Datenverarbeitungs-Optionen) die Option Launch type (Starttyp) aus. b. Wählen Sie unter Launch type (Starttyp) einen Starttyp aus. c. (Optional) Wenn der Fargate Starttyp angegeben ist, geben Sie für Plattformversion die zu verwendende Plattformversion an. Ist keine Plattformversion angegeben, wird die Plattformversion `LATEST` verwendet.

4. Um anzugeben, wie Ihr Service bereitgestellt wird, gehen Sie zum Abschnitt Bereitstellungskonfiguration und wählen Sie dann Ihre Optionen aus.

   a. Belassen Sie für Anwendungstyp die Auswahl auf Service.

b. Wählen Sie für Task definition (Aufgabendefinition) und Revision (Revision) die Aufgabendefinitionsfamilie und die zu verwendende Version aus.

c. Wählen Sie für Servicename, einen Namen für Ihren Service aus.

d. Für Service type (Service-Typ) wählen Sie die Strategie für die Serviceplanung.

   - Wählen Sie Daemon (Daemon), damit der Scheduler genau eine Aufgabe auf jede aktiven Container-Instance verteilt, die alle Bedingungen für die Aufgabenplatzierung erfüllt.
   - Damit der Scheduler die gewünschte Anzahl von Aufgaben in Ihrem Cluster platziert und verwaltet, wählen Sie Replica (Replikat).

e. Wenn Sie Replica (Replikat) gewählt haben, geben Sie bei Desired tasks (Gewünschte Aufgaben) die Anzahl der Aufgaben ein, die im Service gestartet und gepflegt werden sollen.

f. Ermitteln Sie den Bereitstellungstyp für Ihren Service. Erweitern Sie Bereitstellungsoptionen und geben Sie dann die folgenden Parameter an.

Deployment type (Bereitstellungstyp)	Schritte	
Laufende Aktualisierung	a. Für Min running tasks (Min. laufende Aufgaben) geben Sie die untere Grenze für die Anzahl der Aufgaben im Service an, die während einer Bereitstellung in diesem `RUNNING`-Zustand verbleiben müssen, und zwar als Prozentsatz der gewünschten Anzahl von Aufgaben (aufgerundet auf die nächste ganze Zahl). Weitere Informationen finden Sie unter Bereitstellungs-Konfiguration. b. Geben Sie für Max. laufende Aufgaben, die Obergrenze für die Anzahl der Aufgaben im Service ein, die sich während einer Bereitstellung im Status `RUNNING` oder `PENDING` befinden dürfen, und zwar als Prozentsatz der gewünschten Anzahl von Aufgaben (abgerundet auf die nächste Ganzzahl).	

Deployment type (Bereitstellungstyp)	Schritte	
Blau/Grün-Bereitstellung	a. Wählen Sie unter Bereitstellungskonfiguration aus, wie CodeDeploy der Produktionsdatenverkehr während einer Bereitstellung an Ihren Ersatz-Tasksatz weitergeleitet wird. b. Wählen Sie unter Servicerolle für die IAM-Rolle aus CodeDeploy, die der Dienst verwendet, um API-Anfragen an autorisierte AWS-Services Personen zu richten.	

g. Um zu konfigurieren, wie Amazon ECS Bereitstellungsfehler erkennt und behandelt, erweitern Sie Erkennung von Bereitstellungsfehlern, und wählen Sie dann Ihre Optionen.

   i. Um eine Bereitstellung anzuhalten, wenn die Aufgaben nicht gestartet werden können, wählen Sie Use the Amazon ECS deployment circuit breaker (Verwenden des Amazon-ECS-Bereitstellungsschutzschalters).

      Damit die Software die Bereitstellung automatisch auf den Status der letzten abgeschlossenen Bereitstellung zurücksetzt, wenn der Bereitstellungsschutzschalter die Bereitstellung in den Status „Fehlgeschlagen“ versetzt, wählen Sie Rollback bei Fehlern.

   ii. Um eine Bereitstellung auf der Grundlage von Anwendungsmetriken zu beenden, wählen Sie CloudWatch Alarm (en) verwenden aus. Wählen Sie dann unter CloudWatch Alarmname die Alarme aus. Um einen neuen Alarm zu erstellen, gehen Sie zur CloudWatch Konsole.

Damit die Software die Bereitstellung automatisch auf den Status der letzten abgeschlossenen Bereitstellung zurücksetzt, wenn ein CloudWatch Alarm die Bereitstellung in einen fehlgeschlagenen Zustand versetzt, wählen Sie Rollback bei Fehlern.

5. (Optional) Um Service Connect zu verwenden, wählen Sie Turn on Service Connect (Service Connect aktivieren) aus, und geben Sie dann Folgendes an:

   a. Geben Sie unter Service Connect configuration (Service-Connect-Konfiguration) den Client-Modus an.

      - Wenn Ihr Dienst eine Netzwerk-Client-Anwendung ausführt, die nur eine Verbindung zu anderen Diensten im Namespace herstellen muss, wählen Sie Nur Client-Seite.
      - Wenn Ihr Service eine Netzwerk- oder Webservice-Anwendung ausführt und Endpunkte für diesen Service bereitstellen muss und eine Verbindung zu anderen Services im Namespace herstellt, wählen Sie Client and server (Client und Server) aus.

   b. Um einen Namespace zu verwenden, der nicht der Standard-Cluster-Namespace ist, wählen Sie für Namespace den Service-Namespace aus.

   c. (Optional) Wählen Sie die Option Use log collection (Protokollerfassung verwenden), um eine Protokollkonfiguration anzugeben. Für jeden verfügbaren Protokolltreiber gibt es Protokolltreiberoptionen, die angegeben werden müssen. Die Standardoption sendet Container-Logs an CloudWatch Logs. Die anderen Protokolltreiberoptionen werden mit konfiguriert AWS FireLens. Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.

      Im Folgenden wird jedes Container-Protokollziel ausführlicher beschrieben.

      - Amazon CloudWatch — Konfigurieren Sie die Aufgabe, um Container-Logs an CloudWatch Logs zu senden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, mit denen in Ihrem Namen eine CloudWatch Protokollgruppe erstellt wird. Um einen anderen Protokollgruppen-Namen anzugeben, ändern Sie die Werte der Treiberoption.
      - Amazon Data Firehose — Konfigurieren Sie die Aufgabe, um Container-Protokolle an Firehose zu senden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, die Protokolle an einen Firehose-Lieferstream senden. Um einen anderen Namen für den Bereitstellungsdatenstrom anzugeben, ändern Sie die Werte der Treiberoption.

- Amazon Kinesis Data Streams — Konfigurieren Sie die Aufgabe so, dass Container-Logs an Kinesis Data Streams gesendet werden. Es werden die standardmäßigen Protokolltreiberoptionen bereitgestellt, mit denen Protokolle an einen Kinesis Data Streams Streams-Stream gesendet werden. Um einen anderen Datenstrom-Namen anzugeben, ändern Sie die Werte der Treiberoption.
- Amazon OpenSearch Service — Konfigurieren Sie die Aufgabe, um Container-Logs an eine OpenSearch Service-Domain zu senden. Die Optionen für den Protokolltreiber müssen bereitgestellt werden.
- Amazon S3 — Konfigurieren Sie die Aufgabe, um Container-Logs an einen Amazon S3 S3-Bucket zu senden. Die Standardoptionen für den Protokolltreiber sind verfügbar, Sie müssen jedoch einen gültigen Amazon S3 S3-Bucket-Namen angeben.

6. (Optional) Um Service Discovery zu verwenden, wählen Sie Service Discovery verwenden aus und geben Sie dann Folgendes an.

   a. Um einen neuen Namespace zu verwenden, wählen Sie unter Namespace konfigurieren die Option Neuen Namespace erstellen aus, und geben Sie dann einen Namespace-Namen und eine Beschreibung ein. Um einen vorhandenen Namespace zu verwenden, wählen Sie Select an existing namespace und wählen Sie dann den Namespace aus, den Sie verwenden möchten.

   b. Geben Sie Informationen zum Service Discovery-Dienst an, z. B. den Namen und die Beschreibung des Dienstes.

   c. Um Amazon ECS regelmäßige Zustandsprüfungen auf Container-Ebene durchführen zu lassen, wählen Sie Amazon ECS Task Health Propagation aktivieren aus.

   d. Wählen Sie für DNS record type (DNS-Datensatztyp) den DNS-Datensatztyp aus, der für Ihren Service erstellt werden soll. Amazon ECS Service Discovery unterstützt nur A - und SRV-Datensätze, abhängig vom Netzwerkmodus, den Ihre Aufgabendefinition spezifiziert. Informationen über diese Datensatztypen finden Sie unter Unterstützte DNS-Datensatztypen im Amazon Route 53-Entwicklerhandbuch.

      - Wenn die von Ihrer Serviceaufgabe angegebene Aufgabendefinition als Netzwerkmodus `bridge` oder `host` verwendet, werden nur Datensätze vom Typ SRV unterstützt. Wählen Sie eine Kombination aus Containername und -Port aus, die dem Datensatz zugeordnet werden soll.
      - Wenn die Aufgabendefinition, die Ihre Serviceaufgabe angibt, den `awsvpc`-Netzwerkmodus verwendet, wählen Sie entweder den Datensatztyp A oder SRV. Wenn Sie A wählen, fahren Sie mit dem nächsten Schritt fort. Wenn Sie SRV wählen, geben

Sie entweder den Port an, unter dem der Service gefunden werden kann, oder eine Kombination aus Containername und Port, die dem Datensatz zugeordnet werden soll.

Geben Sie für TTL die Zeit in Sekunden ein, für die ein Datensatz von DNS-Resolvern und Webbrowsern zwischengespeichert wird.

7. (Optional) Um einen Load Balancer für Ihren Service zu konfigurieren, erweitern Sie Load balancing (Load Balancing).

   Wählen Sie den Load Balancer aus.

So verwenden Sie diesen Load Balancer	Vorgehensweise
Application Load Balancer	a. Wählen Sie für Load Balancer-Typ Application Load Balancer aus. b. Wählen Sie Erstellen eines neuen Load Balancers, um einen neuen Application Load Balancer zu erstellen, oder Verwenden eines vorhandenen Load Balancers um einen vorhandenen Application Load Balancer auszuwählen. c. Geben Sie für Load balancer name (Name des Load Balancers) einen eindeutigen Namen ein. d. Für Choose container to load balance (Container für den Lastausgleich wählen) wählen Sie den

So verwenden Sie diesen Load Balancer	Vorgehensweise	
	Container aus, der den Service hostet. e. Geben Sie für Listener (Zuhörer) einen Port und ein Protokoll für den Application Load Balancer an, an dem der Application Load Balancer auf Verbindungsanfragen lauschen soll. Standardmäßig wird der Load Balancer so konfiguriert, dass er Port 80 und HTTP verwendet. f. Geben Sie für Target group name (Name der Zielgruppe) einen Namen und ein Protokoll für die Zielgruppe ein, an die der Application Load Balancer Anforderungen weiterleitet. Standardmäßig leitet die Zielgruppe Anforderungen an den ersten Container weiter, der in Ihrer Aufgabendefinition definiert ist. g. Geben Sie unter Verzögerung bei der Registrierung die Anzahl der Sekunden ein, auf die der Load Balancer den Zielstatus ändern	

So verwenden Sie diesen Load Balancer	Vorgehensweise	
	soll. UNUSED Standardmäßig ist ein Zeitraum von 300 Sekunden festgelegt. h. Geben Sie für Health check path (Pfad für die Zustandsprüfung) einen Pfad an, der in Ihrem Container vorhanden ist, an den der Application Load Balancer regelmäßig Anforderungen senden soll, um den Verbindungszustand zwischen dem Application Load Balancer und dem Container zu überprüfen. Der Standard ist das Root-Verzeichnis (/). i. Geben Sie für Health check grace period (Wartefrist der Zustandsprüfungs) die Zeitspanne (in Sekunden) ein, die der Serviceplaner ungesunde Zustandsprüfungen des Ziels des Elastic Load Balancing ignorieren soll.	

So verwenden Sie diesen Load Balancer	Vorgehensweise
Network Load Balancer	a. Wählen Sie für Load balancer type (Load-Balancer-Typ) Network Load Balancer (Network Load Balancer) aus. b. Wählen Sie für Load Balancer (Load Balancer) einen vorhandenen Network Load Balancer. c. Für Choose container to load balance (Container für den Lastausgleich wählen) wählen Sie den Container aus, der den Service hostet. d. Geben Sie unter Target group name (Name der Zielgruppe) einen Namen und ein Protokoll für die Zielgruppe ein, an die der Network Load Balancer Anfragen weiterleitet. Standardmäßig leitet die Zielgruppe Anforderungen an den ersten Container weiter, der in Ihrer Aufgabendefinition definiert ist. e. Geben Sie für Verzögerung bei der Registrierung die Anzahl der Sekunden ein, auf die der Load Balancer den

So verwenden Sie diesen Load Balancer	Vorgehensweise	
	Zielstatus ändern soll. UNUSED Standardm äßig ist ein Zeitraum von 300 Sekunden festgelegt. f. Geben Sie für Health check path (Pfad für die Zustandsprüfung) einen vorhandenen Pfad innerhalb Ihres Container s ein, an den der Network Load Balancer regelmäßi g Anfragen sendet, um den Verbindungszustand zwischen dem Application Load Balancer und dem Container zu überprüfen. Der Standard ist das Root-Verzeichnis (/). g. Geben Sie für Health check grace period (Wartefrist der Zustandsp rüfungs) die Zeitspanne (in Sekunden) ein, die der Serviceplaner ungesunde Zustandsprüfungen des Ziels des Elastic Load Balancing ignorieren soll.	

8. (Optional) Um Service Auto Scaling zu konfigurieren, erweitern Sie Service Auto Scaling und geben Sie dann die folgenden Parameter an.

   a. Um das Auto Scaling der Services zu verwenden, wählen Sie Auto Scaling des Services.

b. Geben Sie für Mindestanzahl von Aufgaben die Untergrenze der Anzahl der Aufgaben ein, die Service Auto Scaling verwenden soll. Die gewünschte Anzahl wird diese Anzahl nicht unterschreiten.

c. Geben Sie für Maximale Anzahl von Aufgaben die Obergrenze für die Anzahl der Aufgaben ein, die Service Auto Scaling verwenden soll. Die gewünschte Anzahl wird diese Anzahl nicht überschreiten.

d. Wählen Sie den Richtlinien-Typ aus. Wählen Sie unter Skalierungsrichtlinientyp eine der folgenden Optionen aus.

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
Zielverfolgung	a. Wählen Sie für Scaling policy type (Typ der Skalierungsrichtlinie) Target tracking (Ziel-Nachverfolgung) aus. b. Geben Sie unter Richtlinienname, einen Namen für diese Richtlinie ein. c. Für ECS-Service-Metrik, wählen Sie eine der folgenden Metriken aus. • ServiceAverageECS-CPUUtilization — Durchschnittliche CPU-Auslastung des Dienstes. • ECS ServiceAverage MemoryUtilization — Durchschnittliche Speicherauslastung des Dienstes. • ALB RequestCount PerTarget — Anzahl der abgeschlossenen Anfragen pro Ziel in einer Application Load Balancer Balancer-Zielgruppe. d. Für Zielwert, geben Sie den Wert ein, den der Service für die	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	ausgewählte Metrik beibehält. e. Geben Sie für Scale-Out -Cooldown-Periode die Zeit in Sekunden ein, die nach einer Scale-Out-Aktivität (Aufgaben hinzufügen) verstrich en sein muss, bevor eine weitere Scale-Out-Aktivität gestartet werden kann. f. Geben Sie für Scale-in-Abklingzeit die Zeit in Sekunden ein, die nach einer Scale-In-Aktivität (Aufgaben entfernen) vergehen muss, bevor eine weitere Scale-In-Aktivität beginnen kann. g. Um zu verhindern, dass die Richtlinie eine Abskalierungsaktiv ität ausführt, wählen Sie Turn off scale-in (Abskalierung ausschalt en). h. • (Optional) Wählen Sie Scale-In deaktivieren aus, wenn Sie möchten, dass Ihre Skalierun gsrichtlinie bei erhöhtem Traffic skaliert wird, sie	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	aber nicht bei sinkendem Traffic skaliert werden muss.	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
Schrittweise Skalierung	a. Wählen Sie für Scaling policy type (Typ der Skalierungsrichtlinie) Step scaling (Schrittweise Skalierung) aus. b. Geben Sie unter Richtliniename den Namen für diese Richtlinie ein. c. Geben Sie unter Alarm name (Alarmname) einen eindeutigen Namen für den Alarm ein. d. Wählen Sie für die Amazon-ECS-Service-Metrik die Metrik, die für den Alarm verwendet werden soll. e. Wählen Sie für Statistik die Alarmstatistik aus. f. Wählen Sie unter Zeitraum den Zeitraum für den Alarm aus. g. Wählen Sie unter Alarmzustand aus, wie die ausgewählte Metrik mit dem definierten Schwellenwert verglichen werden soll. h. Geben Sie unter Schwellenwert für den Vergleich von Metriken	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	und Auswertungszeitraum für die Auslösung des Alarms den Schwellenwert für den Alarm ein und wie lange der Schwellenwert ausgewertet werden soll. i. Führen Sie unter Skalierungsaktionen die folgenden Schritte aus: • Wählen Sie unter Aktion aus, ob Sie eine bestimmte Anzahl für Ihren Service hinzufügen, entfernen oder festlegen möchten. • Wenn Sie Aufgaben hinzufügen oder entfernen möchten, geben Sie unter Wert die Anzahl der Aufgaben (oder den Prozentsatz der vorhandenen Aufgaben) ein, die hinzugefügt oder entfernt werden sollen, wenn die Skalierungsaktion initiiert wird. Wenn Sie die gewünschte Anzahl festlegen möchten,	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	geben Sie die Anzahl der Aufgaben ein. Wählen Sie unter Typ aus, ob der Wert eine Ganzzahl oder ein Prozentwert der vorhandenen gewünschten Anzahl ist. • Geben Sie für Untergrenze und Obergrenze die Untergrenze und Obergrenze der schrittweisen Skalierungsanpassung ein. Standardmäßig ist die untere Grenze für eine Hinzufügerichtlinie die Alarmschwelle und die obere Grenze ist positiv (+) unendlich. Standardmäßig ist die obere Grenze für eine Entfernerichtlinie die Alarmschwelle und die untere Grenze ist negativ (-) unendlich. • (Optional) Fügen Sie zusätzliche Skalierungsoptionen hinzu. Wählen Sie Neue	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	Skalierungsaktion hinzufügen aus, und wiederholen Sie dann die Schritte zur Skalierungsaktion. • Geben Sie unter Abklingzeit die Zeit in Sekunden ein, nach der gewartet werden soll, bis eine vorherige Skalierungsaktivität wirksam wird. Bei einer Hinzufügen-Richtlinie ist dies die Zeit nach einer Scale-Out-Aktivität, in der die Skalierungsrichtlinie Scale-In-Aktivitäten blockiert und begrenzt, wie viele Aufgaben gleichzeitig skaliert werden können. Bei einer Entfernungsrichtlinie ist dies die Zeit nach einer Scale-In-Aktivität, die abgeschlossen sein muss, bevor eine weitere Scale-In-Aktivität beginnen kann.	

9. (Optional) Um eine andere als die standardmäßige Strategie zur Platzierung von Aufgaben zu verwenden, erweitern Sie Task Placement (Platzierung von Aufgaben) und wählen Sie aus den folgenden Optionen aus.

   Weitere Informationen finden Sie unter Wie Amazon ECS Aufgaben auf Container-Instances platziert.

   - AZ Balanced Spread — Verteilen Sie Aufgaben auf Availability Zones und auf Container-Instances in der Availability Zone.
   - AZ Balanced BinPack — Verteilen Sie Aufgaben auf Availability Zones und auf Container-Instances mit dem geringsten verfügbaren Speicher.
   - BinPack— Verteilen Sie Aufgaben auf der Grundlage der geringsten verfügbaren CPU- oder Speichermenge.
   - Eine Aufgabe pro Host — Platzieren Sie maximal eine Aufgabe aus dem Service auf jeder Container-Instance.
   - Benutzerdefiniert — Definieren Sie Ihre eigene Strategie zur Aufgabenverteilung.

   Wenn Sie Custom (Benutzerdefiniert) wählen, definieren Sie den Algorithmus für das Platzieren von Aufgaben und die Regeln, die bei der Aufgabenplatzierung berücksichtigt werden.

   - Unter Strategy (Strategie), für Type (Typ) und Field (Feld), wählen Sie den Algorithmus und die Entität aus, die für den Algorithmus verwendet werden sollen.

     Sie können maximal 5 Strategien angeben.
   - Unter Einschränkung, für Typ und Ausdruck, wählen Sie die Regel und das Attribut für die Einschränkung aus.

     Um beispielsweise die Einschränkung festzulegen, Aufgaben auf T2-Instances zu platzieren, geben Sie für Expression (Ausdruck) attribute:ecs.instance-type =~ t2.* ein.

     Sie können maximal 10 Einschränkungen angeben.
10. Wenn Ihre Aufgabendefinition `awsvpc`-Netzwerkmodus nutzt, erweitern Sie Networking (Netzwerk). Führen Sie die folgenden Schritte aus, um eine benutzerdefinierte Konfiguration anzugeben.

    a. Wählen Sie für VPC die VPC aus, die Sie verwenden möchten.

b. Wählen Sie für Subnets (Subnetze) ein oder mehrere Subnetze in der VPC aus, die der Aufgaben-Scheduler bei der Platzierung Ihrer Aufgaben berücksichtigen soll.

> **Important**
>
> Für den Netzwerkmodus`awsvpc` werden nur private Subnetze unterstützt. Aufgaben erhalten keine öffentlichen IP-Adressen. Daher ist ein NAT-Gateway für ausgehenden Internet-Zugriff erforderlich und eingehender Internetdatenverkehr wird über einen Load Balancer weitergeleitet.

c. Für die VPC-Sicherheitsgruppe können Sie entweder eine vorhandene Sicherheitsgruppe auswählen oder eine neue erstellen. Um eine vorhandene Sicherheitsgruppe zu verwenden, wählen Sie die Sicherheitsgruppe aus und fahren Sie mit dem nächsten Schritt fort. Um eine neue Sicherheitsgruppe zu erstellen, wählen Sie Create a new security group. Sie müssen einen Sicherheitsgruppennamen und eine Beschreibung angeben und dann eine oder mehrere eingehende Regeln für die Sicherheitsgruppe hinzufügen.

11. Wenn Ihre Aufgabe ein Datenvolume verwendet, das mit der Konfiguration bei der Bereitstellung kompatibel ist, können Sie das Volume konfigurieren, indem Sie Volume erweitern.

    Der Datenträgername und der Volumetyp werden konfiguriert, wenn Sie eine Revision der Aufgabendefinition erstellen, und können beim Erstellen eines Dienstes nicht geändert werden. Um den Namen und den Typ des Volumes zu aktualisieren, müssen Sie eine neue Version der Aufgabendefinition erstellen und mithilfe der neuen Version einen Dienst erstellen.

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
Amazon EBS	a. Wählen Sie unter EBS-Volumetyp den Typ des EBS-Volumes aus, den Sie Ihrer Aufgabe zuordnen möchten. b. Geben Sie für Größe (GiB) einen gültigen Wert für die Datenträgergröße in Gibibyte (GiB) ein. Sie können eine Volumengr	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	öße von mindestens 1 GiB und eine maximale Volumengröße von 16.384 GiB angeben. Dieser Wert ist erforderlich, sofern Sie keine Snapshot-ID angeben. c. Geben Sie für IOPS die maximale Anzahl von Eingabe-/Ausgabevo rgängen (IOPS) ein, die das Volume bereitstellen soll. Dieser Wert ist nur für die Volumetypen `io1` `io2`, und konfigurierbar. `gp3` d. Geben Sie für Durchsatz (MIB/s) den Durchsatz in Mebibyte pro Sekunde (MiBpsoder MIB/s) ein, den das Volume bereitste llen soll. Dieser Wert ist nur für den Volumetyp konfigurierbar. `gp3` e. Wählen Sie für Snapshot-ID einen vorhandenen Amazon EBS-Volum e-Snapshot aus oder geben Sie den ARN eines Snapshots ein, wenn Sie ein Volume aus einem Snapshot erstellen möchten. Sie können auch ein neues, leeres Volume	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	erstellen, indem Sie keine Snapshot-ID auswählen oder eingeben. f. Wählen Sie unter Dateisystemtyp den Typ des Dateisystems aus, das für das Speichern und Abrufen von Daten auf dem Volume verwendet werden soll. Sie können entweder den Betriebss ystemstandard oder einen bestimmten Dateisyst emtyp wählen. Die Standardeinstellung für Linux ist`XFS`. Für Volumes, die aus einem Snapshot erstellt wurden, müssen Sie denselben Dateisystemtyp angeben, den das Volume bei der Erstellung des Snapshots verwendet hat. Wenn der Dateisystemtyp nicht übereinstimmt, kann die Aufgabe nicht gestartet werden. g. Wählen Sie für die Infrastrukturrolle eine IAM-Rolle mit den erforderlichen Berechtig ungen, die es Amazon ECS ermöglichen,	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Amazon EBS-Volumes für Aufgaben zu verwalten. Sie können die `AmazonECSInfrastructureRolePolicyForVolumes` verwaltete Richtlinie an die Rolle anhängen, oder Sie können die Richtlinie als Leitfaden verwenden, um eine eigene Richtlinie mit Berechtigungen zu erstellen und anzuhängen, die Ihren spezifischen Anforderungen entsprechen. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter IAM-Rolle für die Amazon ECS-Infrastruktur. h. Wählen Sie unter Verschlüsselung die Option Standard, wenn Sie die Amazon EBS-Verschlüsselung standardmäßig verwenden möchten. Wenn für Ihr Konto standardmäßig Verschlüsselung konfiguriert ist, wird das Volume mit dem Schlüssel AWS Key Management Service (AWS KMS) verschlüs	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	selt, der in der Einstellu ng angegeben ist. Wenn Sie Standard wählen und die Amazon EBS-Standardverschlüsselung nicht aktiviert ist, wird das Volume unverschlüsselt. Wenn Sie Benutzerd efiniert wählen, können Sie eine Option Ihrer Wahl für AWS KMS key die Volumenverschlüsselung angeben. Wenn Sie „Keine“ wählen, wird das Volume unverschlüsselt, es sei denn, Sie haben die Verschlüsselung standardmäßig konfiguri ert oder Sie erstellen ein Volume aus einem verschlüsselten Snapshot. i. Wenn Sie Benutzerd efiniert für Verschlüs selung ausgewählt haben, müssen Sie angeben AWS KMS key , welche Sie verwenden möchten. Wählen Sie für KMS-Schlüssel einen Schlüssel -ARN aus AWS KMS key oder geben Sie	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	einen ein. Wenn Sie Ihr Volume mithilfe eines symmetrischen, vom Kunden verwalteten Schlüssels verschlüs seln möchten, stellen Sie sicher, dass Sie in Ihrer AWS KMS key Richtlini e über die richtigen Berechtigungen verfügen. Weitere Informationen finden Sie unter Datenvers chlüsselung für Amazon EBS-Volumes. j. (Optional) Unter Tags können Sie Ihrem Amazon EBS-Volume Tags hinzufügen, indem Sie entweder Tags aus der Aufgabendefinition oder dem Service weitergeb en oder indem Sie Ihre eigenen Tags angeben. Wenn Sie Tags aus der Aufgabendefinition weitergeben möchten, wählen Sie Aufgabend efinition für Tags weitergeben aus. Wenn Sie Tags aus dem Service weitergeben möchten, wählen Sie Service für die Weitergabe von Tags aus.	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Wenn Sie Nicht weitergeben oder wenn Sie keinen Wert auswählen, werden die Tags nicht weitergegeben. Wenn Sie Ihre eigenen Tags angeben möchten, wählen Sie Tag hinzufügen und geben Sie dann den Schlüssel und den Wert für jedes Tag ein, das Sie hinzufügen. Weitere Informationen zum Taggen von Amazon EBS-Volumes finden Sie unter Tagging Amazon EBS-Volumes.	

12. (Optional) Um Ihren Service und Ihre Aufgaben besser identifizieren zu können, erweitern Sie den Bereich Tags (Tags) und konfigurieren Sie dann Ihre Tags.

    Damit Amazon ECS alle neu gestarteten Aufgaben automatisch mit Tags für den Clusternamen und den Aufgabendefinitions-Tags versieht, wählen Sie Verwaltete Amazon-ECS-Tags einschalten und dann für Tags verbreiten von die Option Aufgabendefinitionen.

    Damit Amazon ECS alle neu gestarteten Aufgaben automatisch mit Tags für den Clustername und die Service-Tags versieht, wählen Sie Verwaltete Amazon-ECS-Tags einschalten und dann für Tags verbreiten von die Option Service.

    Hinzufügen oder Entfernen eines Tag.

    - [Ein Tag hinzufügen] Wählen Sie Add tag (Tag hinzufügen) und führen Sie dann das Folgende aus:
      - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.

- Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
- [Tag entfernen] Wählen Sie neben dem Tag die Option Remove tag (Tag löschen) aus.

Aktualisieren eines Amazon ECS-Service mithilfe der Konsole

Sie können einen Amazon-ECS-Service mit der Amazon-ECS-Konsole aktualisieren. Die aktuelle Service-Konfiguration ist vorausgefüllt. Sie können die Aufgabendefinition, die gewünschte Anzahl an Aufgaben, die Kapazitätsanbieterstrategie, die Plattformversion und die Bereitstellungskonfiguration oder eine beliebige Kombination hiervon aktualisieren.

Informationen zum Aktualisieren der Blau/Grün-Bereitstellungs-Konfiguration finden Sie unter Aktualisierung einer Blue/Green-Bereitstellung von Amazon ECS mithilfe der Konsole.

Die folgenden Hinweise gelten für die Verwendung der Konsole:

Wenn Sie Ihren Service vorübergehend beenden möchten, setzen Sie Gewünschte Aufgaben auf 0. Wenn Sie dann bereit sind, den Dienst zu starten, aktualisieren Sie den Dienst mit der ursprünglichen Anzahl der gewünschten Aufgaben.

Die folgenden Hinweise gelten für die Verwendung der Konsole:

- Sie müssen den verwenden AWS Command Line Interface , um einen Dienst zu aktualisieren, der einen der folgenden Parameter verwendet:
  - Blau/Grün-Bereitstellungen
  - Service Discovery — Sie können nur Ihre Service Discovery-Konfiguration anzeigen.
  - Tracking-Richtlinie mit einer benutzerdefinierten Metrik
  - Service aktualisieren — Sie können die `awsvpc` Netzwerkkonfiguration und den Kulanzzeitraum für die Integritätsprüfung nicht aktualisieren.

  Informationen zum Aktualisieren eines Dienstes mithilfe von finden Sie update-servicein der AWS Command Line Interface Referenz. AWS CLI
- Wenn Sie die von Containern verwendeten Ports in einer Aufgabendefinition ändern, müssen Sie möglicherweise die Sicherheitsgruppen für die Container-Instances aktualisieren, damit diese mit den aktualisierten Ports arbeiten können.
- Amazon ECS aktualisiert die Sicherheitsgruppen nicht automatisch, die mit Elastic Load Balancing-Load Balancern oder Amazon-ECS-Container-Instances verbunden sind.

- Wenn Ihr Service einen Load Balancer verwendet, kann die Konfiguration des Load Balancer, die für Ihren Service bei seiner Erstellung definiert wurde, nicht mit der Konsole geändert werden. Sie können stattdessen das AWS CLI oder SDK verwenden, um die Load Balancer-Konfiguration zu ändern. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz.
- Wenn Sie die Aufgabendefinition für den Service aktualisieren, müssen der Containername und der Container-Port, die in der Load-Balancer-Konfiguration angegeben sind, in der Aufgabendefinition verbleiben.

Sie können einen vorhandenen Service während seiner Ausführung aktualisieren, um einige der Serverkonfigurationsparameter zu ändern, etwa die Anzahl der vom Service verwalteten Aufgaben, welche Aufgabendefinition von den Aufgaben verwendet wird, oder auch die Plattformversion, wenn Ihre Aufgaben den Starttyp Fargate verwenden. Ein Service, der eine Linux-Plattformversion verwendet, kann nicht aktualisiert werden, um eine Windows-Plattformversion zu verwenden und umgekehrt. Wenn Sie eine Anwendung haben, die mehr Kapazität benötigt, können Sie Ihren Service skalieren. Wenn Sie über ungenutzte Kapazitäten verfügen, können Sie die Anzahl der gewünschten Aufgaben in Ihrem Service reduzieren und Ressourcen freigeben und herunterskalieren.

Wenn Sie ein aktualisiertes Container-Image für Ihre Aufgaben verwenden möchten, können Sie eine neue Aufgabendefinitionsvision mit diesem Abbild erstellen und es mithilfe der Option Force new deployment (Neue Bereitstellung erzwingen) in der Konsole für Ihren Service bereitstellen.

Der Service Scheduler verwendet die minimalen gesunden Prozent und maximalen Prozentparameter (in der Bereitstellungskonfiguration für den Service), um die Bereitstellungsstrategie festzulegen.

Wenn ein Service den Bereitstellungstyp laufende Aktualisierung (`ECS`) verwendet, stellen die minimalen gesunden Prozent eine untere Grenze für die Anzahl der Aufgaben in einem Service dar, die während einer Bereitstellung im Zustand `RUNNING` verbleiben müssen, als Prozentsatz der gewünschten Anzahl von Aufgaben (aufgerundet auf die nächste ganze Zahl). Der Parameter gilt auch, wenn sich Container-Instances im Zustand `DRAINING` befinden, wenn der Service Aufgaben mit dem Starttyp EC2 enthält. Sie können diesen Parameter verwenden, um die Bereitstellung ohne zusätzliche Cluster-Kapazität durchzuführen. Wenn Ihr Service z. B. eine gewünschte Anzahl von vier Aufgaben und einen Mindestprozentsatz an gesunden Aufgaben von 50 % hat, könnte der Planer zwei bestehende Aufgaben stoppen, um Cluster-Kapazität freizugeben, bevor er zwei neue Aufgaben startet. Aufgaben für Services, die keinen Load Balancer verwenden, werden als fehlerfrei betrachtet, wenn sie sich im Status `RUNNING` befinden. Aufgaben für Services, die einen Load

Balancer verwenden, gelten als gesund, wenn sie sich im Zustand `RUNNING` befinden und vom Load Balancer als gesund gemeldet werden. Der Standardwert für den minimalen gesunden Prozentsatz ist 100 Prozent.

Wenn ein Service den Bereitstellungstyp laufende Aktualisierung (`ECS`) verwendet, stellt der Parameter maximale Prozent eine Obergrenze für die Anzahl der Aufgaben in einem Service dar, die im Zustand `PENDING`, `RUNNING` oder `STOPPING` während einer Bereitstellung zulässig sind, und zwar als Prozentsatz der gewünschten Anzahl von Aufgaben (abgerundet auf die nächste ganze Zahl). Der Parameter gilt auch, wenn sich Container-Instances im Zustand `DRAINING` befinden, wenn der Service Aufgaben mit dem Starttyp EC2 enthält. Mithilfe dieses Parameters können Sie die Größe der Bareitstellungsstapel definieren. Wenn Ihr Service beispielsweise eine gewünschte Anzahl von vier Aufgaben und einen maximalen Prozentwert von 200 Prozent hat, kann der Planer vier neue Aufgaben starten, bevor er die vier älteren Aufgaben stoppt. Voraussetzung dafür ist, dass die dafür erforderlichen Cluster-Ressourcen verfügbar sind. Der Standardwert für den maximalen Prozentsatz beträgt 200 Prozent.

Wenn der Service Scheduler während einer Aktualisierung eine Aufgabe ersetzt, entfernt der Service zuerst die Aufgabe aus dem Load Balancer (falls verwendet) und wartet, bis die Verbindungen ausgelaufen sind. Dann wird das Äquivalent von docker stop an die Container ausgegeben, die in der Aufgabe ausgeführt werden. Das löst ein `SIGTERM`-Signal und eine Zeitbeschränkung von 30 Sekunden aus, nach der `SIGKILL` gesendet und ein Stoppen der Container erzwungen wird. Wenn der Container das `SIGTERM`-Signal normal verarbeitet und innerhalb von 30 Sekunden nach Erhalt des Signals schließt, wird das `SIGKILL`-Signal gesendet. Der Service-Scheduler startet und stoppt Aufgaben entsprechend der Definition in Ihren Einstellungen für den mindestens fehlerfreien Prozentsatz und den maximalen Prozentsatz.

Der Service-Scheduler ersetzt auch Aufgaben, die nach einem Fehlschlagen einer Container-Zustandsprüfung oder einer Load-Balancer-Zielgruppen-Zustandsprüfung als fehlerhaft eingestuft wurden. Dieser Ersatz hängt von den Parametern `maximumPercent` und `desiredCount` der Servicedefinition ab. Wenn eine Aufgabe als fehlerhaft markiert ist, startet der Service-Scheduler zunächst eine Ersatzaufgabe. Dann passiert Folgendes.

- Wenn die Ersatzaufgabe den Integritätsstatus hat`HEALTHY`, stoppt der Service Scheduler die fehlerhafte Aufgabe
- Wenn die Ersatzaufgabe den Zustand `UNHEALTHY` hat, stoppt der Scheduler entweder die fehlerhafte Ersatzaufgabe oder die vorhandene fehlerhafte Aufgabe, sodass die Gesamtanzahl der Aufgaben auf einen Wert gleich `desiredCount` eingestellt wird.

Wenn der Parameter `maximumPercent` den Scheduler daran hindert, zuerst eine Ersatzaufgabe zu starten, stoppt der Scheduler fehlerhafte Aufgaben einzeln nach dem Zufallsprinzip, um Kapazität freizugeben, und startet dann eine Ersatzaufgabe. Der Start- und Stoppvorgang wird fortgesetzt, bis alle fehlerhaften Aufgaben durch fehlerfreie Aufgaben ersetzt wurden. Sobald alle fehlerhaften Aufgaben ersetzt wurden und nur noch fehlerfreie Aufgaben ausgeführt werden, werden, wenn die Gesamtzahl der Aufgaben `desiredCount` übersteigt, die fehlerfreien Aufgaben nach dem Zufallsprinzip angehalten, bis die Gesamtzahl der Aufgaben gleich `desiredCount` ist. Weitere Informationen zu `maximumPercent` und `desiredCount` finden Sie unter Servicedefinitionsparamater.

> **Important**
>
> Wenn Sie die von den Containern in einer Aufgabendefinition verwendeten Ports ändern, müssen Sie eventuell die Sicherheitsgruppen für die Container-Instances aktualisieren, damit diese mit den aktualisierten Ports funktionieren.
> Wenn Sie die Aufgabendefinition für den Service aktualisieren, müssen der Containername und der Container-Port, die bei der Erstellung des Services angegeben wurden, in der Aufgabendefinition bleiben.
> Amazon ECS aktualisiert die Sicherheitsgruppen nicht automatisch, die mit Elastic Load Balancing-Load Balancern oder Amazon-ECS-Container-Instances verbunden sind.

So aktualisieren Sie einen Service (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Cluster-Seite den Cluster aus.
3. Aktivieren Sie auf der Seite mit den Clusterdetails im Abschnitt Dienste das Kontrollkästchen neben dem Dienst, und wählen Sie dann Aktualisieren aus.
4. Wählen Sie Force new deployment (Neue Bereitstellung erzwingen) aus, damit Ihr Service eine neue Bereitstellung startet.
5. Wählen Sie für Aufgabendefinition die Aufgabendefinitionsfamilie und die Version aus.

   > **Important**
   >
   > Die Konsole überprüft, ob die ausgewählte Aufgabendefinitionsfamilie und Revision mit der definierten Rechenkonfiguration kompatibel sind. Wenn Sie eine Warnung erhalten,

überprüfen Sie sowohl die Kompatibilität Ihrer Aufgabendefinition als auch die von Ihnen ausgewählte Rechenkonfiguration.

6. Geben Sie unter Gewünschte Aufgaben die Anzahl der Aufgaben ein, die Sie für den Dienst ausführen möchten.
7. Für Min running tasks (Min. laufende Aufgaben) geben Sie die untere Grenze für die Anzahl der Aufgaben im Service an, die während einer Bereitstellung in diesem `RUNNING`-Zustand verbleiben müssen, und zwar als Prozentsatz der gewünschten Anzahl von Aufgaben (aufgerundet auf die nächste ganze Zahl). Weitere Informationen finden Sie unter Bereitstellungs-Konfiguration.
8. Geben Sie für Max. laufende Aufgaben, die Obergrenze für die Anzahl der Aufgaben im Service ein, die sich während einer Bereitstellung im Status `RUNNING` oder `PENDING` befinden dürfen, und zwar als Prozentsatz der gewünschten Anzahl von Aufgaben (abgerundet auf die nächste Ganzzahl).
9. Um zu konfigurieren, wie Amazon ECS Bereitstellungsfehler erkennt und behandelt, erweitern Sie Erkennung von Bereitstellungsfehlern, und wählen Sie dann Ihre Optionen.

    a. Um eine Bereitstellung anzuhalten, wenn die Aufgaben nicht gestartet werden können, wählen Sie Use the Amazon ECS deployment circuit breaker (Verwenden des Amazon-ECS-Bereitstellungsschutzschalters).

    Damit die Software die Bereitstellung automatisch auf den Status der letzten abgeschlossenen Bereitstellung zurücksetzt, wenn der Bereitstellungsschutzschalter die Bereitstellung in den Status „Fehlgeschlagen“ versetzt, wählen Sie Rollback bei Fehlern.

    b. Um eine Bereitstellung auf der Grundlage von Anwendungsmetriken zu beenden, wählen Sie CloudWatch Alarm (en) verwenden aus. Wählen Sie dann unter CloudWatch Alarmname die Alarme aus. Um einen neuen Alarm zu erstellen, gehen Sie zur CloudWatch Konsole.

    Damit die Software die Bereitstellung automatisch auf den Status der letzten abgeschlossenen Bereitstellung zurücksetzt, wenn ein CloudWatch Alarm die Bereitstellung in einen fehlgeschlagenen Zustand versetzt, wählen Sie Rollback bei Fehlern.
10. Um die Rechenoptionen zu ändern, erweitern Sie Compute-Konfiguration und gehen Sie dann wie folgt vor:

    a. Wählen Sie für Dienste auf AWS Fargate, für Plattformversion, die neue Version aus.

b. Gehen Sie für Dienste, die eine Kapazitätsanbieterstrategie verwenden, unter Kapazitätsanbieterstrategie wie folgt vor:

- Um einen zusätzlichen Kapazitätsanbieter hinzuzufügen, wählen Sie Weitere hinzufügen aus. Wählen Sie dann für Kapazitätsanbieter den Kapazitätsanbieter aus.
- Um einen Kapazitätsanbieter zu entfernen, wählen Sie rechts neben dem Kapazitätsanbieter die Option Entfernen aus.

Ein Dienst, der einen Auto Scaling Scaling-Gruppenkapazitätsanbieter verwendet, kann nicht aktualisiert werden, um einen Fargate-Kapazitätsanbieter zu verwenden. Ein Dienst, der einen Fargate-Kapazitätsanbieter verwendet, kann nicht aktualisiert werden, um einen Auto Scaling Scaling-Gruppenkapazitätsanbieter zu verwenden.

11. (Optional) Um Service Auto Scaling zu konfigurieren, erweitern Sie Service Auto Scaling und geben Sie dann die folgenden Parameter an.

a. Um das Auto Scaling der Services zu verwenden, wählen Sie Auto Scaling des Services.

b. Geben Sie für Mindestanzahl von Aufgaben die Untergrenze der Anzahl der Aufgaben ein, die Service Auto Scaling verwenden soll. Die gewünschte Anzahl wird diese Anzahl nicht unterschreiten.

c. Geben Sie für Maximale Anzahl von Aufgaben die Obergrenze für die Anzahl der Aufgaben ein, die Service Auto Scaling verwenden soll. Die gewünschte Anzahl wird diese Anzahl nicht überschreiten.

d. Wählen Sie den Richtlinien-Typ aus. Wählen Sie unter Skalierungsrichtlinientyp eine der folgenden Optionen aus.

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
Zielverfolgung	a. Wählen Sie für Scaling policy type (Typ der Skalierungsrichtlinie) Target tracking (Ziel-Nachverfolgung) aus.	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	b. Geben Sie unter Richtlinienname, einen Namen für diese Richtlinie ein. c. Für ECS-Service-Metrik, wählen Sie eine der folgenden Metriken aus. • ServiceAverageECS-CPUUtilization — Durchschnittliche CPU-Auslastung des Dienstes. • ECS ServiceAverage MemoryUtilization — Durchschnittliche Speicherauslastung des Dienstes. • ALB RequestCount PerTarget — Anzahl der abgeschlossenen Anfragen pro Ziel in einer Application Load Balancer Balancer-Zielgruppe. d. Für Zielwert, geben Sie den Wert ein, den der Service für die ausgewählte Metrik beibehält. e. Geben Sie für Scale-Out-Cooldown-Periode die Zeit in Sekunden ein, die nach einer Scale-	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	Out-Aktivität (Aufgaben hinzufügen) verstrich en sein muss, bevor eine weitere Scale-Out-Aktivität gestartet werden kann. f. Geben Sie für Scale-in-Abklingzeit die Zeit in Sekunden ein, die nach einer Scale-In-Aktivität (Aufgaben entfernen) vergehen muss, bevor eine weitere Scale-In-Aktivität beginnen kann. g. Um zu verhindern, dass die Richtlinie eine Abskalierungsaktiv ität ausführt, wählen Sie Turn off scale-in (Abskalierung ausschalt en). h. • (Optional) Wählen Sie Scale-In deaktivieren aus, wenn Sie möchten, dass Ihre Skalierun gsrichtlinie bei erhöhtem Traffic skaliert wird, sie aber nicht bei sinkendem Traffic skaliert werden muss.	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
Schrittweise Skalierung	a. Wählen Sie für Scaling policy type (Typ der Skalierungsrichtlinie) Step scaling (Schrittweise Skalierung) aus. b. Geben Sie unter Richtliniename den Namen für diese Richtlinie ein. c. Geben Sie unter Alarm name (Alarmname) einen eindeutigen Namen für den Alarm ein. d. Wählen Sie für die Amazon-ECS-Service-Metrik die Metrik, die für den Alarm verwendet werden soll. e. Wählen Sie für Statistik die Alarmstatistik aus. f. Wählen Sie unter Zeitraum den Zeitraum für den Alarm aus. g. Wählen Sie unter Alarmzustand aus, wie die ausgewählte Metrik mit dem definierten Schwellenwert verglichen werden soll. h. Geben Sie unter Schwellenwert für den Vergleich von Metriken	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	und Auswertungszeitraum für die Auslösung des Alarms den Schwellenwert für den Alarm ein und wie lange der Schwellenwert ausgewertet werden soll. i. Führen Sie unter Skalierungsaktionen die folgenden Schritte aus: • Wählen Sie unter Aktion aus, ob Sie eine bestimmte Anzahl für Ihren Service hinzufügen, entfernen oder festlegen möchten. • Wenn Sie Aufgaben hinzufügen oder entfernen möchten, geben Sie unter Wert die Anzahl der Aufgaben (oder den Prozentsatz der vorhandenen Aufgaben) ein, die hinzugefügt oder entfernt werden sollen, wenn die Skalierungsaktion initiiert wird. Wenn Sie die gewünschte Anzahl festlegen möchten,	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	geben Sie die Anzahl der Aufgaben ein. Wählen Sie unter Typ aus, ob der Wert eine Ganzzahl oder ein Prozentwert der vorhandenen gewünschten Anzahl ist. • Geben Sie für Untergrenze und Obergrenze die Untergrenze und Obergrenze der schrittweisen Skalierungsanpassung ein. Standardmäßig ist die untere Grenze für eine Hinzufügerichtlinie die Alarmschwelle und die obere Grenze ist positiv (+) unendlich. Standardmäßig ist die obere Grenze für eine Entfernerichtlinie die Alarmschwelle und die untere Grenze ist negativ (-) unendlich. • (Optional) Fügen Sie zusätzliche Skalierungsoptionen hinzu. Wählen Sie Neue	

Zur Verwendung dieser Richtlinie …	Vorgehensweise	
	Skalierungsaktion hinzufügen aus, und wiederholen Sie dann die Schritte zur Skalierungsaktion. • Geben Sie unter Abklingzeit die Zeit in Sekunden ein, nach der gewartet werden soll, bis eine vorherige Skalierungsaktivität wirksam wird. Bei einer Hinzufügen-Richtlinie ist dies die Zeit nach einer Scale-Out-Aktivität, in der die Skalierungsrichtlinie Scale-In-Aktivitäten blockiert und begrenzt, wie viele Aufgaben gleichzeitig skaliert werden können. Bei einer Entfernungsrichtlinie ist dies die Zeit nach einer Scale-In-Aktivität, die abgeschlossen sein muss, bevor eine weitere Scale-In-Aktivität beginnen kann.	

12. (Optional) Um Service Connect zu verwenden, wählen Sie Turn on Service Connect (Service Connect aktivieren) aus, und geben Sie dann Folgendes an:

a. Geben Sie unter Service Connect configuration (Service-Connect-Konfiguration) den Client-Modus an.

   - Wenn Ihr Service eine Netzwerk-Client-Anwendung ausführt, die nur eine Verbindung zu anderen Services im Namespace herstellen muss, wählen Sie Client side only (Nur Client-Seite) aus.
   - Wenn Ihr Service eine Netzwerk- oder Webservice-Anwendung ausführt und Endpunkte für diesen Service bereitstellen muss und eine Verbindung zu anderen Services im Namespace herstellt, wählen Sie Client and server (Client und Server) aus.

b. Um einen Namespace zu verwenden, der nicht der Standard-Cluster-Namespace ist, wählen Sie für Namespace den Service-Namespace aus.

13. Wenn Ihre Aufgabe ein Datenvolume verwendet, das mit der Konfiguration bei der Bereitstellung kompatibel ist, können Sie das Volume konfigurieren, indem Sie Volume erweitern.

    Der Datenträgername und der Volumetyp werden konfiguriert, wenn Sie eine Revision der Aufgabendefinition erstellen, und können nicht geändert werden, wenn Sie einen Dienst aktualisieren. Um den Namen und den Typ des Volumes zu aktualisieren, müssen Sie eine neue Version der Aufgabendefinition erstellen und den Dienst mithilfe der neuen Version aktualisieren.

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
Amazon EBS	a. Wählen Sie unter EBS-Volumetyp den Typ des EBS-Volumes aus, den Sie Ihrer Aufgabe zuordnen möchten. b. Geben Sie für Größe (GiB) einen gültigen Wert für die Datenträgergröße in Gibibyte (GiB) ein. Sie können eine Volumengröße von mindestens 1 GiB und eine maximale Volumengröße von 16.384 GiB angeben. Dieser Wert	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	ist erforderlich, sofern Sie keine Snapshot-ID angeben. c. Geben Sie für IOPS die maximale Anzahl von Eingabe-/Ausgabevorgängen (IOPS) ein, die das Volume bereitstellen soll. Dieser Wert ist nur für die Volumetypen `io1` `io2`, und konfigurierbar. `gp3` d. Geben Sie für Durchsatz (MIB/s) den Durchsatz in Mebibyte pro Sekunde (MiBpsoder MIB/s) ein, den das Volume bereitstellen soll. Dieser Wert ist nur für den Volumetyp konfigurierbar. `gp3` e. Wählen Sie für Snapshot-ID einen vorhandenen Amazon EBS-Volume-Snapshot aus oder geben Sie den ARN eines Snapshots ein, wenn Sie ein Volume aus einem Snapshot erstellen möchten. Sie können auch ein neues, leeres Volume erstellen, indem Sie keine Snapshot-ID auswählen oder eingeben.	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	f. Wählen Sie unter Dateisystemtyp den Typ des Dateisystems aus, das für das Speichern und Abrufen von Daten auf dem Volume verwendet werden soll. Sie können entweder den Betriebss ystemstandard oder einen bestimmten Dateisyst emtyp wählen. Die Standardeinstellung für Linux ist`XFS`. Für Volumes, die aus einem Snapshot erstellt wurden, müssen Sie denselben Dateisystemtyp angeben, den das Volume bei der Erstellung des Snapshots verwendet hat. Wenn der Dateisystemtyp nicht übereinstimmt, kann die Aufgabe nicht gestartet werden. g. Wählen Sie für die Infrastrukturrolle eine IAM-Rolle mit den erforderlichen Berechtig ungen, die es Amazon ECS ermöglichen, Amazon EBS-Volum es für Aufgaben zu verwalten. Sie können die	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	`AmazonECSInfrastructureRolePolicyForVolumes` verwaltete Richtlinie an die Rolle anhängen, oder Sie können die Richtlinie als Leitfaden verwenden, um eine eigene Richtlinie mit Berechtigungen zu erstellen und anzuhängen, die Ihren spezifischen Anforderungen entsprechen. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter IAM-Rolle für die Amazon ECS-Infrastruktur. h. Wählen Sie unter Verschlüsselung die Option Standard, wenn Sie die Amazon EBS-Verschlüsselung standardmäßig verwenden möchten. Wenn für Ihr Konto standardmäßig Verschlüsselung konfiguriert ist, wird das Volume mit dem Schlüssel AWS Key Management Service (AWS KMS) verschlüsselt, der in der Einstellung angegeben ist. Wenn Sie Standard wählen	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	und die Amazon EBS-Standardverschlüsselung nicht aktiviert ist, wird das Volume unverschlüsselt. Wenn Sie Benutzerdefiniert wählen, können Sie eine Option Ihrer Wahl für AWS KMS key die Volumenverschlüsselung angeben. Wenn Sie „Keine“ wählen, wird das Volume unverschlüsselt, es sei denn, Sie haben die Verschlüsselung standardmäßig konfiguriert oder Sie erstellen ein Volume aus einem verschlüsselten Snapshot. i. Wenn Sie Benutzerdefiniert für Verschlüsselung ausgewählt haben, müssen Sie angeben AWS KMS key , welche Sie verwenden möchten. Wählen Sie für KMS-Schlüssel einen Schlüssel-ARN aus AWS KMS key oder geben Sie einen ein. Wenn Sie Ihr Volume mithilfe eines symmetrischen, vom	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	Kunden verwalteten Schlüssels verschlüsseln möchten, stellen Sie sicher, dass Sie in Ihrer AWS KMS key Richtlinie über die richtigen Berechtigungen verfügen. Weitere Informationen finden Sie unter Datenverschlüsselung für Amazon EBS-Volumes. j. (Optional) Unter Tags können Sie Ihrem Amazon EBS-Volume Tags hinzufügen, indem Sie entweder Tags aus der Aufgabendefinition oder dem Service weitergeben oder indem Sie Ihre eigenen Tags angeben. Wenn Sie Tags aus der Aufgabendefinition weitergeben möchten, wählen Sie Aufgabendefinition für Tags weitergeben aus. Wenn Sie Tags aus dem Service weitergeben möchten, wählen Sie Service für die Weitergabe von Tags aus. Wenn Sie Nicht weitergeben oder wenn Sie keinen Wert auswählen, werden	

Um diesen Volumetyp zu konfigurieren	Vorgehensweise	
	die Tags nicht weitergeg eben. Wenn Sie Ihre eigenen Tags angeben möchten, wählen Sie Tag hinzufüge n und geben Sie dann den Schlüssel und den Wert für jedes Tag ein, das Sie hinzufügen. Weitere Informationen zum Taggen von Amazon EBS-Volumes finden Sie unter Tagging Amazon EBS-Volumes.	

14. (Optional) Um Ihren Service leichter identifizieren zu können, erweitern Sie die Tags (Tags) und konfigurieren Sie dann Ihre Tags.
    - [Tag hinzufügen] Wählen Sie Tag hinzufügen und gehen Sie wie folgt vor:
        - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
        - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
    - [Tag entfernen] Wählen Sie neben dem Tag die Option Remove tag (Tag löschen) aus.
15. Wählen Sie Aktualisieren.

Aktualisierung einer Blue/Green-Bereitstellung von Amazon ECS mithilfe der Konsole

Sie können eine Blau/Grün-Bereitstellungs-Konfiguration mithilfe der Amazon-ECS-Konsole aktualisieren. Die aktuelle Blau/Grün-Bereitstellungs-Konfiguration ist vorausgefüllt. Sie können die folgenden Optionen für Blau/Grün-Bereitstellungen aktualisieren:

- Name der Bereitstellungsgruppe — Die CodeDeploy Bereitstellungseinstellungen

- Anwendungsname — Die CodeDeploy Bereitstellungsgruppe
- Bereitstellungskonfiguration — Wie CodeDeploy leitet der Produktionsdatenverkehr während einer Bereitstellung an Ihren Ersatz-Tasksatz weiter
- Test-Listener auf dem Load Balancer — CodeDeploy verwendet den Test-Listener, um Ihren Testdatenverkehr während einer Bereitstellung an den Ersatz-Tasksatz weiterzuleiten

Sie müssen die neue Option konfigurieren, bevor Sie die Konfiguration aktualisieren.

So aktualisieren Sie eine Blau/Grün-Bereitstellungs-Konfiguration (Amazon-ECS-Konsole)

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Cluster-Seite den Cluster aus.
3. Wählen Sie auf der Seite Cluster overview (Cluster-Übersicht) den Service und dann Update (Aktualisieren) aus.
4. Erweitern Sie Bereitstellungsoptionen — Bereitgestellt von und wählen Sie dann aus CodeDeploy, welche Optionen aktualisiert werden sollen:
   - Um die CodeDeploy Bereitstellungsgruppe zu ändern, wählen Sie unter Anwendungsname die Bereitstellungsgruppe aus.
   - Um die CodeDeploy Bereitstellungseinstellungen zu ändern, wählen Sie unter Name der Bereitstellungsgruppe die Gruppe aus.
   - Um zu ändern, CodeDeploy wie der Produktionsdatenverkehr während einer Bereitstellung an Ihren Ersatz-Tasksatz weitergeleitet wird, wählen Sie unter Bereitstellungskonfiguration die Option aus.
5. Wählen Sie die Lebenszyklus-Hooks der Bereitstellung und die zugehörigen Lambda-Funktionen aus, die im Rahmen der neuen Version der Service-Bereitstellung ausgeführt werden sollen. Die verfügbaren Lebenszyklus-Hooks sind:
   - BeforeInstall— Verwenden Sie diesen Event-Hook für den Bereitstellungslebenszyklus, um eine Lambda-Funktion aufzurufen, bevor der Ersatzaufgabensatz erstellt wird. Das Ergebnis der Lambda-Funktion bei diesem Lebenszyklusereignis löst keinen Rücksetzvorgang aus.
   - AfterInstall— Verwenden Sie diesen Event-Hook für den Bereitstellungslebenszyklus, um eine Lambda-Funktion aufzurufen, nachdem der Ersatzaufgabensatz erstellt wurde. Das Ergebnis der Lambda-Funktion bei diesem Lebenszyklusereignis kann einen Rücksetzvorgang auslösen.

- BeforeAllowVerkehr — Verwenden Sie diesen Event-Hook für den Bereitstellungslebenszyklus, um eine Lambda-Funktion aufzurufen, bevor der Produktionsdatenverkehr an den Ersatz-Tasksatz umgeleitet wurde. Das Ergebnis der Lambda-Funktion bei diesem Lebenszyklusereignis kann einen Rücksetzvorgang auslösen.
- AfterAllowVerkehr — Verwenden Sie diesen Event-Hook für den Bereitstellungslebenszyklus, um eine Lambda-Funktion aufzurufen, nachdem der Produktionsdatenverkehr an den Ersatz-Tasksatz umgeleitet wurde. Das Ergebnis der Lambda-Funktion bei diesem Lebenszyklusereignis kann einen Rücksetzvorgang auslösen.

6. Um den Test-Listener zu ändern, erweitern Sie Load balancing und wählen Sie dann für Test Listener for CodeDeploy Deployment den Test-Listener aus.
7. Wählen Sie Aktualisieren.

Löschen eines Amazon ECS-Service mithilfe der Konsole

Sie können einen Amazon-ECS-Service über die Konsole löschen. Vor dem Löschen wird der Service automatisch auf Null herunterskaliert. Load-Balancer-Ressourcen oder Serviceerkennungen, die mit dem Service verbunden sind, sind von der Löschung des Service nicht betroffen. Informationen zum Löschen Ihrer Elastic Load Balancing-Ressourcen finden Sie je nach Ihrem Load Balancer-Typ in einem der folgenden Themen: Löschen eines Application Load Balancers oder Löschen eines Network Load Balancers.

Wenn Sie einen Service löschen und noch Aufgaben laufen, für die eine Bereinigung erforderlich ist, wechselt der Servicestatus von `ACTIVE` zu `DRAINING` und der Service ist in der Konsole oder im `ListServices` API-Vorgang nicht mehr sichtbar. Nachdem alle Aufgaben entweder in den Status `STOPPING` oder `STOPPED` übergegangen ist, ändert sich der Servicestatus von `DRAINING` zu `INACTIVE`. Dienste im `INACTIVE` Status `DRAINING` oder können während des `DescribeServices` API-Vorgangs weiterhin angezeigt werden.

> **Important**
>
> Wenn Sie versuchen, einen neuen Dienst mit demselben Namen wie ein vorhandener Dienst im `DRAINING` Status `ACTIVE` oder zu erstellen, erhalten Sie eine Fehlermeldung.

Verfahren

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.

2. Wählen Sie auf der Cluster-Seite den Cluster für den Service aus.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Services aus.
5. Wählen Sie die Services und dann Delete (Löschen) aus.
6. Um einen Service zu löschen, auch wenn er nicht auf null Aufgaben reduziert wurde, wählen Sie die Option Force delete service (Löschen des Services erzwingen).
7. Geben Sie in der Bestätigungsaufforderung Löschen ein und wählen Sie dann Löschen aus.

Stellen Sie Amazon ECS-Services bereit, indem Sie Aufgaben ersetzen

Wenn Sie einen Service erstellen, der den Bereitstellungstyp Rolling Update (`ECS`) verwendet, ersetzt der Amazon ECS-Service Scheduler die aktuell ausgeführten Aufgaben durch neue Aufgaben. Die Anzahl der Aufgaben, die Amazon ECS während einer fortlaufenden Aktualisierung für den Service hinzufügt oder entfernt, wird durch die Service-Bereitstellungskonfiguration gesteuert. Die Bereitstellungskonfiguration besteht aus Folgendem:

- `minimumHealthyPercent` stellt die Untergrenze für die Anzahl der Aufgaben dar, die für einen Service während einer Bereitstellung oder wenn eine Container-Instance entleert wird, als Prozentsatz der gewünschten Anzahl von Aufgaben für den Service. Dieser Wert wird aufgerundet. Zum Beispiel, wenn der minimale gesunde Prozentsatz `50` ist und die gewünschte Aufgabenanzahl vier ist, kann der Scheduler zwei bestehende Aufgaben stoppen, bevor er zwei neue Aufgaben startet. Ebenso kann der Scheduler, wenn der minimale fehlerfreie Prozentsatz 75 % beträgt und die gewünschte Anzahl zwei ist, keine Aufgaben stoppen, da der resultierende Wert auch zwei ist.

  Wenn Aufgaben fehlerhaft werden, startet der Amazon ECS-Service Scheduler zuerst Ersatzaufgaben und behält Aufgaben bei, bis die `minimumHealthyPercent` Ersatzaufgaben fehlerfrei sind. Wenn die Ersatzaufgaben gestartet werden und wieder fehlerfrei sind, werden die fehlerhaften Aufgaben nach und nach gestoppt.
- `maximumPercent` stellt die Obergrenze für die Anzahl der Aufgaben dar, die für einen Service während einer Bereitstellung oder wenn eine Container-Instance entleert wird, als Prozentsatz der gewünschten Anzahl von Aufgaben für einen Service. Dieser Wert wird abgerundet. Zum Beispiel, wenn der maximale Prozentsatz `200` ist und die gewünschte Aufgabenanzahl vier ist, kann der Scheduler vier neue Aufgaben starten, bevor er vier vorhandene Aufgaben beendet. Ebenso, wenn der maximale Prozentsatz `125`. ist und die gewünschte Aufgabenanzahl drei ist, kann der Scheduler keine Aufgaben starten, da der resultierende Wert ebenfalls drei ist.

> **Important**
>
> Beim Festlegen eines minimalen fehlerfreien Prozentsatzes oder eines maximalen Prozentsatzes sollten Sie sicherstellen, dass der Planer mindestens eine Aufgabe anhalten oder starten kann, wenn eine Bereitstellung initiiert wird. Wenn Ihr Service über eine Bereitstellung verfügt, die aufgrund einer ungültigen Bereitstellungskonfiguration nicht mehr besteht, wird eine Serviceereignismeldung gesendet. Weitere Informationen finden Sie unter Service (*service-name*) konnte Tasks während einer Bereitstellung aufgrund der Konfiguration der Dienstbereitstellung nicht anhalten oder starten. Aktualisieren Sie den Wert minimumHealthyPercent oder MaximumPercent und versuchen Sie es erneut..

Bei einer fortlaufenden Bereitstellung wird anhand des Bereitstellungsschutzschalters festgestellt, ob die Aufgaben einen stabilen Zustand erreichen. Der Bereitstellungsschutzschalter kann eine Bereitstellung bei einem Fehler optional zurücksetzen.

Erkennung von Fehlern

Es gibt zwei Methoden, mit denen Sie schnell feststellen können, wann eine Bereitstellung fehlgeschlagen ist, und dann optional den Fehler auf die letzte funktionierende Bereitstellung zurücksetzen können.

- the section called “ Wie der Deployment Circuit Breaker Ausfälle erkennt”
- the section called “Wie CloudWatch Alarme Bereitstellungsfehler erkennen”

Die Methoden können getrennt oder zusammen verwendet werden. Wenn beide Methoden verwendet werden, wird die Bereitstellung als fehlgeschlagen eingestuft, sobald das Fehlerkriterium für eine der beiden Fehlermethoden erfüllt ist.

Bestimmen Sie anhand der folgenden Anleitungen, welche Methode verwendet werden soll:

- Schutzschalter – Verwenden Sie diese Methode, wenn Sie eine Bereitstellung anhalten möchten, falls die Aufgaben nicht gestartet werden können.
- CloudWatch Alarme — Verwenden Sie diese Methode, wenn Sie eine Bereitstellung auf der Grundlage von Anwendungsmetriken beenden möchten.

So erkennt der Amazon ECS Deployment Circuit Breaker Fehler

Beim Bereitstellungsschutzschalter handelt es sich um einen Mechanismus zur fortlaufenden Aktualisierung, der feststellt, ob die Aufgaben einen stabilen Status erreichen. Der Bereitstellungsschutzschalter verfügt über eine Option, mit der eine fehlgeschlagene Bereitstellung automatisch auf die Bereitstellung zurückgesetzt wird, die sich im Status `COMPLETED` befindet.

Wenn sich der Status einer Servicebereitstellung ändert, sendet Amazon ECS ein Ereignis zur Änderung des Status der Servicebereitstellung an EventBridge. Dies bietet eine programmgesteuerte Möglichkeit, den Status Ihrer Servicebereitstellungen zu überwachen. Weitere Informationen finden Sie unter Ereignisse zur Änderung des Bereitstellungsstatus des Amazon ECS-Service. Wir empfehlen Ihnen, eine EventBridge Regel mit dem Wert `eventName` von zu erstellen und zu überwachen, `SERVICE_DEPLOYMENT_FAILED` sodass Sie manuelle Maßnahmen ergreifen können, um Ihre Bereitstellung zu starten. Weitere Informationen finden Sie unter EventBridgeRegel erstellen im EventBridge Amazon-Benutzerhandbuch.

Wenn der Bereitstellungsschutzschalter feststellt, dass eine Bereitstellung fehlgeschlagen ist, sucht er nach der letzten Bereitstellung, die sich im Status `COMPLETED` befindet. Dies ist die Bereitstellung, die als Rollback-Bereitstellung verwendet wird. Wenn das Rollback beginnt, ändert sich die Bereitstellung von `COMPLETED` zu `IN_PROGRESS`. Das bedeutet, dass für die Bereitstellung kein weiterer Rollback in Frage kommt, bis der Status `COMPLETED` erreicht wurde. Wenn der Bereitstellungsschutzschalter keine Bereitstellung findet, die sich in im Status `COMPLETED` befindet, startet der Schutzschalter keine neuen Aufgaben und die Bereitstellung wird angehalten.

Wenn Sie einen Service erstellen, verfolgt der Scheduler die Aufgaben, die nicht gestartet werden konnten, in zwei Phasen.

- Phase 1 — Der Scheduler überwacht die Aufgaben, um festzustellen, ob sie in den Status RUNNING übergehen.
  - Erfolgreich — Es besteht die Möglichkeit, dass die Bereitstellung in den Status ABGESCHLOSSEN übergeht, da mehrere Aufgaben in den Status RUNNING übergegangen sind. Die Fehlerkriterien werden übersprungen und der Schutzschalter wechselt zur Stufe 2.
  - Fehler — Es gibt aufeinanderfolgende Aufgaben, die nicht in den Status RUNNING übergegangen sind, und die Bereitstellung könnte in den Status FAILED übergehen.
- Phase 2 — Die Bereitstellung geht in diese Phase über, wenn sich mindestens eine Aufgabe im Status RUNNING befindet. Der Schutzschalter überprüft die Integritätsprüfungen für die Aufgaben in der aktuellen Bereitstellung, die gerade ausgewertet werden. Bei den validierten

Zustandsprüfungen handelt es sich um Elastic Load Balancing, AWS Cloud Map Service Health Checks und Container Health Checks.

- Erfolgreich — Es gibt mindestens eine Aufgabe im Status Running, deren Integritätsprüfungen bestanden wurden.
- Fehlgeschlagen — Die Aufgaben, die aufgrund fehlgeschlagener Integritätsprüfungen ersetzt wurden, haben den Schwellenwert für Fehler erreicht.

Beachten Sie Folgendes, wenn Sie die Deployment Circuit Breaker-Methode für einen Dienst verwenden. EventBridge generiert die Regel.

- Die `DescribeServices`-Antwort bietet Einblick in den Status einer Bereitstellung, den `rolloutState` und `rolloutStateReason`. Wenn eine neue Bereitstellung gestartet wird, beginnt der Rollout-Status in einem `IN_PROGRESS`-Zustand. Wenn der Service einen stabilen Zustand erreicht, wechselt der Rollout-Status zu `COMPLETED`. Wenn der Service keinen stabilen Status erreicht und der Schutzschalter aktiviert wird, wechselt die Bereitstellung in einen `FAILED`-Status. Eine Bereitstellung in einem `FAILED`-Status startet keine neuen Aufgaben.
- Zusätzlich zu den Ereignissen zur Statusänderung der Servicebereitstellung, die Amazon ECS für gestartete und abgeschlossene Bereitstellungen sendet, sendet Amazon ECS auch ein Ereignis, wenn eine Bereitstellung mit aktiviertem Schutzschalter fehlschlägt. Diese Ereignisse enthalten Details dazu, warum eine Bereitstellung fehlgeschlagen ist oder ob eine Bereitstellung aufgrund eines Rollbacks gestartet wurde. Weitere Informationen finden Sie unter Ereignisse zur Änderung des Bereitstellungsstatus des Amazon ECS-Service.
- Wenn eine neue Bereitstellung gestartet wird, weil eine vorherige Bereitstellung fehlgeschlagen ist und ein Rollback aufgetreten ist, zeigt das `reason`-Feld des Ereignisses für die Statusänderung der Service-Bereitstellung an, dass die Bereitstellung aufgrund eines Rollbacks gestartet wurde.
- Der Bereitstellungsschutzschalter wird nur für Amazon-ECS-Services unterstützt, die den Bereitstellungscontroller für fortlaufende Aktualisierung (`ECS`) verwenden.
- Sie müssen die Amazon ECS-Konsole oder den, AWS CLI wenn Sie den Deployment Circuit Breaker mit der CloudWatch Option verwenden, verwenden. Weitere Informationen finden Sie unter the section called “Erstellen eines Services mit definierten Parametern” und create-service in der AWS Command Line Interface -Referenz.

Das folgende `create-service` AWS CLI Beispiel zeigt, wie ein Linux-Service erstellt wird, wenn der Deployment Circuit Breaker mit der Rollback-Option verwendet wird.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration "deploymentCircuitBreaker={enable=true,rollback=true}"
 \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Beispiel:

Bereitstellung 1 befindet sich im Status `COMPLETED`.

Bereitstellung 2 kann nicht gestartet werden, sodass der Schutzschalter per Rollback auf Bereitstellung 1 zurückkehrt. Bereitstellung 1 wechselt in den Status `IN_PROGRESS`.

Bereitstellung 3 wird gestartet und es gibt keine Bereitstellung im Status `COMPLETED`, sodass Bereitstellung 3 kein Rollback oder Starten von Aufgaben durchführen kann.

Fehlerschwellenwert

Der Bereitstellungs-Leistungsschalter berechnet den Schwellenwert und verwendet dann den Wert, um zu bestimmen, wann die Bereitstellung in einen `FAILED`-Zustand versetzt werden soll.

Der Auslöseschalter hat einen Mindestschwellenwert von 3 und einen Höchstschwellenwert von 200 und verwendet die Werte in der folgenden Formel, um den Bereitstellungsfehler zu ermitteln.

```
Minimum threshold <= 0.5 * desired task count => maximum threshold
```

Wenn das Ergebnis der Berechnung größer als der Mindestwert von 3, aber kleiner als der Höchstwert von 200 ist, wird der Ausfallschwellenwert auf den berechneten Schwellenwert (aufgerundet) gesetzt.

> **Note**
>
> Sie können keinen der Schwellenwerte ändern.

Es gibt zwei Phasen für die Überprüfung des Bereitstellungsstatus.

1. Der Bereitstellungs-Leistungsschalter überwacht Aufgaben, die Teil der Bereitstellung sind und sucht nach Aufgaben, die sich im `RUNNING`-Zustand befinden. Der Scheduler ignoriert die Fehlerkriterien, wenn sich eine Aufgabe in der aktuellen Bereitstellung im `RUNNING`-Zustand befindet und fährt mit der nächsten Stufe fort. Wenn Aufgaben den `RUNNING`-Zustand nicht erreichen, erhöht der Bereitstellungs-Leistungsschalter die Fehleranzahl um eins. Wenn die Fehleranzahl dem Schwellenwert entspricht, wird die Bereitstellung als `FAILED` markiert.
2. Diese Phase tritt ein, wenn der `RUNNING` Status eine oder mehrere Aufgaben enthält. Der Bereitstellungs-Leistungsschalter führt Zustandsprüfungen der folgenden Ressourcen für die Aufgaben in der aktuellen Bereitstellung durch:
   - Elastic Load Balancing-Load Balancer
   - AWS Cloud Map Dienst
   - Zustandsprüfungen für Amazon-ECS-Container

   Wenn eine Zustandsprüfung für die Aufgabe fehlschlägt, erhöht der Bereitstellungs-Leistungsschalter die Fehleranzahl um eins. Wenn die Fehleranzahl dem Schwellenwert entspricht, wird die Bereitstellung als `FAILED` markiert.

Die folgende Tabelle bietet einige Beispiele.

Gewünschte Aufgabenanzahl	Berechnung	Threshold
1	`3 <= 0.5 * 1 => 200`	3 (der berechnete Wert liegt unter dem Minimum)
25	`3 <= 0.5 * 25 => 200`	13 (Der Wert wird aufgerundet)
400	`3 <= 0.5 * 400 => 200`	200
800	`3 <= 0.5 * 800 => 200`	200 (Der berechnete Wert ist größer als das Maximum)

Wenn der Schwellenwert beispielsweise 3 ist, startet der Schutzschalter mit der Fehleranzahl, die auf 0 gesetzt ist. Wenn eine Aufgabe den `RUNNING` Status nicht erreicht, erhöht der

Auslöseschutzschalter die Anzahl der Fehler um eins. Wenn die Anzahl der Fehler gleich 3 ist, wird die Bereitstellung als `FAILED` markiert.

Weitere Beispiele zur Verwendung der Rollback-Option finden Sie unter Ankündigung des Amazon-ECS-Bereitstellungsschutzschalters.

So erkennen CloudWatch Alarme Bereitstellungsfehler bei Amazon ECS

Sie können Amazon ECS so konfigurieren, dass die Bereitstellung auf „Fehlgeschlagen“ gesetzt wird, wenn erkannt wird, dass ein bestimmter CloudWatch Alarm in den `ALARM` Status übergegangen ist.

Sie können die Konfiguration optional so einrichten, dass eine fehlgeschlagene Bereitstellung auf die zuletzt abgeschlossene Bereitstellung zurückgesetzt wird.

Das folgende `create-service` AWS CLI Beispiel zeigt, wie ein Linux-Service erstellt wird, wenn die Bereitstellungsalarme mit der Rollback-Option verwendet werden.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration
 "alarms={alarmNames=[alarm1Name,alarm2Name],enable=true,rollback=true}" \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Beachten Sie Folgendes, wenn Sie die CloudWatch Amazon-Alarmmethode für einen Service verwenden.

- Die Bake-Zeit ist ein Zeitraum nach dem Aufskalieren einer neuen Serviceversion und dem Abskalieren der alten Serviceversion. Während dieser Zeit überwacht Amazon ECS weiterhin den der Bereitstellung zugeordneten Alarm. Amazon ECS berechnet diesen Zeitraum basierend auf der Alarmkonfiguration, die der Bereitstellung zugeordnet ist.
- Der `deploymentConfiguration`-Anfrageparameter enthält jetzt den `alarms`-Datentyp. Sie können die Alarmnamen angeben, festlegen, ob die Methode verwendet werden soll und ob ein Rollback initiiert werden soll, wenn die Alarme einen Bereitstellungsfehler anzeigen. Weitere Informationen finden Sie CreateServicein der Amazon Elastic Container Service API-Referenz.

- Die `DescribeServices`-Antwort bietet Einblick in den Status einer Bereitstellung, den `rolloutState` und `rolloutStateReason`. Wenn eine neue Bereitstellung gestartet wird, beginnt der Rollout-Status in einem `IN_PROGRESS`-Status. Wenn der Service einen stabilen Status erreicht und die Bake-Zeit abgeschlossen ist, wechselt der Rollout-Status zu `COMPLETED`. Wenn der Service keinen stabilen Status erreicht und der Alarm in den `ALARM`-Status übergegangen ist, wechselt die Bereitstellung in einen `FAILED`-Status. Eine Bereitstellung in einem `FAILED`-Status startet keine neuen Aufgaben.
- Zusätzlich zu den Ereignissen zur Statusänderung der Service-Bereitstellung, die Amazon ECS für gestartete und abgeschlossene Bereitstellungen sendet, sendet Amazon ECS auch ein Ereignis, wenn eine Bereitstellung mit aktiviertem Alarmen fehlschlägt. Diese Ereignisse enthalten Details dazu, warum eine Bereitstellung fehlgeschlagen ist oder ob eine Bereitstellung aufgrund eines Rollbacks gestartet wurde. Weitere Informationen finden Sie unter Ereignisse zur Änderung des Bereitstellungsstatus des Amazon ECS-Service.
- Wenn eine neue Bereitstellung gestartet wird, weil eine vorherige Bereitstellung fehlgeschlagen ist und ein Rollback aufgetreten ist, zeigt das `reason`-Feld des Ereignisses für die Statusänderung der Service-Bereitstellung an, dass die Bereitstellung aufgrund eines Rollbacks gestartet wurde.
- Wenn Sie den Deployment Circuit Breaker und die CloudWatch Amazon-Alarme verwenden, um Fehler zu erkennen, können beide einen Bereitstellungsfehler auslösen, sobald die Kriterien für eine der beiden Methoden erfüllt sind. Ein Rollback erfolgt, wenn Sie die Rollback-Option für die Methode verwenden, die den Bereitstellungsfehler ausgelöst hat.
- Die CloudWatch Amazon-Alarme werden nur für Amazon ECS-Services unterstützt, die den Rolling Update (`ECS`) Deployment Controller verwenden.
- Sie können diese Option mithilfe der Amazon ECS-Konsole oder der konfigurieren AWS CLI. Weitere Informationen finden Sie unter the section called “Erstellen eines Services mit definierten Parametern” und create-service in der AWS Command Line Interface -Referenz.
- Möglicherweise stellen Sie fest, dass der Bereitstellungsstatus über einen längeren Zeitraum `IN_PROGRESS` bleibt. Der Grund dafür ist, dass Amazon ECS den Status erst ändert, wenn die aktive Bereitstellung gelöscht wurde. Dies geschieht erst nach der Bake-Zeit. Abhängig von Ihrer Alarmkonfiguration scheint die Bereitstellung möglicherweise einige Minuten länger zu dauern als ohne Verwendung von Alarmen (obwohl der neue primäre Aufgabensatz hochskaliert und die alte Bereitstellung herunterskaliert wird). Wenn Sie CloudFormation Timeouts verwenden, sollten Sie erwägen, die Timeouts zu erhöhen. Weitere Informationen finden Sie unter Erstellen von Wartebedingungen in einer Vorlage im AWS CloudFormation -Benutzerhandbuch.
- Amazon ECS ruft `DescribeAlarms` auf, um die Alarme abzufragen. Die Anrufe werden auf die mit Ihrem `DescribeAlarms` Konto verknüpften CloudWatch Servicekontingente angerechnet.

Wenn Sie über andere AWS Dienste verfügen, die anrufen`DescribeAlarms`, kann dies Auswirkungen auf die Abfrage der Alarme durch Amazon ECS haben. Wenn beispielsweise ein anderer Service genügend `DescribeAlarms` Anrufe tätigt, um das Kontingent zu erreichen, wird dieser Service gedrosselt und Amazon ECS ist ebenfalls gedrosselt und kann keine Alarme abfragen. Wenn während des Drosselungszeitraums ein Alarm ausgelöst wird, übersieht Amazon ECS den Alarm möglicherweise und das Rollback findet möglicherweise nicht statt. Es gibt keine weiteren Auswirkungen auf die Bereitstellung. Weitere Informationen zu CloudWatch Servicekontingenten finden Sie unter CloudWatch Servicekontingenten im CloudWatch Benutzerhandbuch.

- Wenn sich ein Alarm zu Beginn einer Bereitstellung im Status `ALARM` befindet, überwacht Amazon ECS keine Alarme für die Dauer dieser Bereitstellung (Amazon ECS ignoriert die Alarmkonfiguration). Dieses Verhalten eignet sich für den Fall, dass Sie eine neue Bereitstellung starten möchten, um einen Fehler bei der ersten Bereitstellung zu beheben.

Empfohlene Alarme

Wir empfehlen die Verwendung der folgenden Alarmmetriken:

- Wenn Sie einen Application Load Balancer verwenden, verwenden Sie die Application-Load-Balancer-Metriken `HTTPCode_ELB_5XX_Count` und `HTTPCode_ELB_4XX_Count`. Diese Metriken überprüfen, ob HTTP-Spitzen vorliegen. Weitere Informationen zu den Application Load Balancer-Metriken finden Sie unter CloudWatch Metriken für Ihren Application Load Balancer im Benutzerhandbuch für Application Load Balancers.
- Verwenden Sie bei einer vorhandenen Anwendung die `CPUUtilization`- und `MemoryUtilization`-Metriken. Diese Metriken überprüfen den Prozentsatz von CPU und Arbeitsspeicher, den der Cluster oder Service verwendet. Weitere Informationen finden Sie unter the section called “Überlegungen”.
- Wenn Sie Amazon Simple Queue Service Warteschlangen in Ihren Aufgaben verwenden, verwenden Sie die `ApproximateNumberOfMessagesNotVisible` Amazon SQS-Metrik. Diese Metrik prüft die Anzahl der Mitteilungen in der Warteschlange, die verzögert und nicht sofort zum Lesen verfügbar sind. Weitere Informationen zu Amazon SQS-Metriken finden Sie unter Verfügbare CloudWatch Metriken für Amazon SQS im Amazon Simple Queue Service Developer Guide.

Überprüfen Sie den Status eines Amazon ECS-Service vor der Bereitstellung

Der Bereitstellungstyp Blau/Grün verwendet das Bereitstellungsmodell Blau/Grün, das von gesteuert wird. CodeDeploy Mit diesem Bereitstellungstyp können Sie eine neue Bereitstellung eines Service überprüfen, bevor Sie den Produktionsverkehr an ihn senden. Weitere Informationen finden Sie unter Was ist CodeDeploy im AWS CodeDeploy Benutzerhandbuch. Überprüfen Sie den Status eines Amazon ECS-Service vor der Bereitstellung

Es gibt drei Möglichkeiten, wie sich der Verkehr während einer blauen/grünen Bereitstellung verlagern kann:

- Kanarisch — Der Verkehr wird in zwei Schritten verlagert. Sie können aus vordefinierten Canary-Optionen auswählen, die den Prozentsatz des Datenverkehrs, der im ersten Inkrementschritt zu Ihrem aktualisierten Aufgabensatz verschoben wird, sowie das Zeitintervall (in Minuten) angeben, das verstreichen soll, bevor der restliche Datenverkehr im zweiten Inkrementschritt verschoben wird.
- Linear — Der Verkehr wird in gleichen Schritten verschoben, wobei zwischen den einzelnen Schritten die gleiche Anzahl von Minuten liegt. Sie können aus vordefinierten linearen Optionen auswählen, die den Prozentsatz des Datenverkehrs, der in den einzelnen Inkrementschritten verschoben wird, sowie das Zeitintervall (in Minuten) zwischen den einzelnen Inkrementschritten angeben.
- Alles auf einmal — Der gesamte Datenverkehr wird auf einmal vom ursprünglichen Task-Set auf das aktualisierte Task-Set übertragen.

Die folgenden Komponenten werden von Amazon ECS verwendet CodeDeploy , wenn ein Service den Bereitstellungstyp Blau/Grün verwendet:

CodeDeploy Anwendung

Eine Sammlung von CodeDeploy Ressourcen. Sie besteht aus einer oder mehreren Bereitstellungsgruppen.

CodeDeploy Bereitstellungsgruppe

Die Bereitstellungseinstellungen. Diese bestehen aus:

- Amazon-ECS-Cluster und -Service
- Load Balancer Zielgruppen- und Listener-Informationen

- Rollback-Strategie für die Bereitstellung
- Einstellungen zur Verkehrsumlenkung
- Beendigungseinstellungen der ursprünglichen Revision
- Bereitstellungskonfiguration
- CloudWatch Alarm-Konfiguration, die so eingerichtet werden kann, dass Bereitstellungen gestoppt werden
- SNS- oder CloudWatch Events-Einstellungen für Benachrichtigungen

Weitere Informationen finden Sie unter Arbeiten mit Bereitstellungsgruppen im AWS CodeDeploy - Benutzerhandbuch.

CodeDeploy Konfiguration der Bereitstellung

Gibt an, CodeDeploy wie der Produktionsdatenverkehr während einer Bereitstellung an Ihren Ersatz-Tasksatz weitergeleitet wird. Die folgenden vordefinierten linearen und Canary-Bereitstellungskonfigurationen sind verfügbar. Sie können auch benutzerdefinierte lineare und Canary-Bereitstellungen erstellen. Weitere Informationen finden Sie unter Arbeiten mit Bereitstellungskonfigurationen im AWS CodeDeploy -Benutzerhandbuch.

- CodeDeployDefault.ecs AllAt Once: Verschiebt den gesamten Datenverkehr auf einmal in den aktualisierten Amazon ECS-Container
- CodeDeploydefault.ecsLinear10 PercentEvery 1Minutes: Verschiebt jede Minute 10 Prozent des Datenverkehrs, bis der gesamte Verkehr verlagert ist.
- CodeDeploydefault.ecsLinear10 PercentEvery 3Minutes: Verschiebt 10 Prozent des Datenverkehrs alle 3 Minuten, bis der gesamte Verkehr verlagert ist.
- CodeDeploydefault.ecscanary10Percent5Minutes: Verschiebt 10 Prozent des Datenverkehrs in der ersten Stufe. Die restlichen 90 Prozent werden fünf Minuten später bereitgestellt.
- CodeDeploydefault.ecscanary10Percent15Minutes: Verschiebt 10 Prozent des Datenverkehrs in der ersten Stufe. Die restlichen 90 Prozent werden 15 Minuten später bereitgestellt.

Revision

Eine Revision ist die Anwendungsspezifikationsdatei (Datei). CodeDeploy AppSpec In der AppSpec Datei geben Sie den vollständigen ARN der Aufgabendefinition sowie den Container und Port Ihres Ersatz-Tasksatzes an, an den der Datenverkehr weitergeleitet werden soll, wenn eine neue Bereitstellung erstellt wird. Der Containername muss einer der Containernamen sein, auf die in Ihrer Aufgabendefinition verwiesen wird. Wenn die Netzwerkkonfiguration oder die Plattformversion in der Dienstdefinition aktualisiert wurde, müssen Sie diese Details auch in der

AppSpec Datei angeben. Sie können auch die Lambda-Funktionen angeben, die während des Bereitstellungs-Lebenszyklus ausgeführt werden sollen. Mit den Lambda-Funktionen können Sie während der Bereitstellung Tests durchführen und Metriken zurückgeben. Weitere Informationen finden Sie unter AppSpec Dateireferenz im AWS CodeDeploy Benutzerhandbuch.

Überlegungen

Beachten Sie Folgendes, wenn Sie den Blau/Grün-Bereitstellungstyp verwenden:

- Wenn ein Amazon-ECS-Service mit dem Blau/Grün-Bereitstellungstyp erstmals erstellt wird, wird ein Amazon-ECS-Aufgabensatz erstellt.
- Sie müssen den Service für die Verwendung von entweder einem Application Load Balancer oder Network Load Balancer konfigurieren. Nachfolgend sind die Anforderungen an den Load Balancer aufgeführt:
  - Sie müssen dem Load Balancer, der zur Weiterleitung des Produktionsverkehrs verwendet wird, einen Produktions-Listener hinzufügen.
  - Dem Load Balancer, der zur Weiterleitung des Testverkehrs verwendet wird, kann ein optionaler Test-Listener hinzugefügt werden. Wenn Sie einen Test-Listener angeben, CodeDeploy leitet er Ihren Testdatenverkehr während einer Bereitstellung an den Ersatz-Tasksatz weiter.
  - Sowohl der Produktions- als auch der Test-Listener müssen demselben Load Balancer angehören.
  - Sie müssen für den Load Balancer eine Zielgruppe definieren. Die Zielgruppe leitet den Datenverkehr über den Produktions-Listener an die ursprüngliche Aufgabe weiter, die in einem Service festgelegt wurde.
  - Bei Verwendung eines Network Load Balancer wird nur die `CodeDeployDefault.ECSAllAtOnce`-Bereitstellungskonfiguration unterstützt.
- Bei Services, die für die Verwendung des Auto Scaling des Services und der blau/grüne Bereitstellungstyp konfiguriert sind, wird das Auto Scaling während einer Bereitstellung nicht blockiert, die Bereitstellung kann jedoch unter Umständen fehlschlagen. Im Folgenden wird dieses Verhalten ausführlich beschrieben.
  - Wenn ein Service skaliert wird und eine Bereitstellung gestartet wird, wird das grüne Task-Set erstellt. Es CodeDeploy wird bis zu einer Stunde gewartet, bis das grüne Task-Set den stabilen Status erreicht hat. Bis dahin wird kein Traffic verlagert.
  - Wenn sich ein Service gerade in einer blau/grünen Bereitstellung befindet und ein Skalierungsereignis auftritt, wird der Datenverkehr für 5 Minuten weiter verschoben. Wenn

der Dienst nicht innerhalb von 5 Minuten den Steady-State erreicht, CodeDeploy wird die Bereitstellung gestoppt und als fehlgeschlagen markiert.

- Wenn sich ein Service gerade in einer blau/grünen Bereitstellung befindet und ein Skalierungsereignis auftritt, wird die gewünschte Aufgabenzahl möglicherweise auf einen unerwarteten Wert festgelegt. Dies wird durch das Auto Scaling verursacht, die die Anzahl der laufenden Aufgaben als aktuelle Kapazität berücksichtigt, was der doppelten Anzahl von Aufgaben entspricht, die in der Berechnung der gewünschten Aufgabenzahl verwendet werden.

- Aufgaben, die den Starttyp Fargate oder die Controller-Typen `CODE_DEPLOY` verwenden, unterstützen die `DAEMON`-Scheduling-Strategie nicht.
- Wenn Sie zum ersten Mal eine CodeDeploy Anwendung und eine Bereitstellungsgruppe erstellen, müssen Sie Folgendes angeben:
  - Sie müssen zwei Zielgruppen für den Load Balancer definieren. Eine Zielgruppe ist die ursprüngliche Zielgruppe, die für den Load Balancer beim Anlegen des Amazon-ECS-Service definiert wurde. Die einzige Anforderung der zweiten Zielgruppe besteht darin, dass sie nicht mit einem anderen Load Balancer als dem, den der Service verwendet, verknüpft werden darf.
- Wenn Sie eine CodeDeploy Bereitstellung für einen Amazon ECS-Service erstellen, CodeDeploy wird in der Bereitstellung ein Ersatzaufgabensatz (oder ein grüner Tasksatz) erstellt. Wenn Sie dem Load Balancer einen Test-Listener hinzugefügt haben, CodeDeploy leitet er Ihren Testdatenverkehr an den Ersatz-Aufgabensatz weiter. Dies ist der Zeitpunkt, an dem Sie Validierungstests durchführen können. Leitet CodeDeploy dann den Produktionsdatenverkehr entsprechend den Einstellungen für die Umleitung des Datenverkehrs für die Bereitstellungsgruppe vom ursprünglichen Task-Set zum Ersatz-Task-Set um.

Erforderliche IAM-Berechtigungen

Blaue/grüne Bereitstellungen werden durch eine Kombination aus Amazon ECS und CodeDeploy APIs ermöglicht. Benutzer müssen über die entsprechenden Berechtigungen für diese Services verfügen, bevor sie Blue/Green-Bereitstellungen von Amazon ECS in den AWS Management Console oder mit den AWS CLI oder SDKs verwenden können.

Zusätzlich zu den IAM-Standardberechtigungen für das Erstellen und Aktualisieren von Services erfordert Amazon ECS folgende Berechtigungen. Diese Berechtigungen wurden der `AmazonECS_FullAccess`-IAM-Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AmazonECS_ FullAccess.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:StopDeployment",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetApplicationRevision",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:BatchGetApplicationRevisions",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetApplications",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ContinueDeployment",
                "sns:ListTopics",
                "cloudwatch:DescribeAlarms",
                "lambda:ListFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

> **Note**
>
> Zusätzlich zu den standardmäßigen Amazon-ECS-Berechtigungen, die zum Ausführen von Aufgaben und Services erforderlich sind, benötigen Benutzer auch `iam:PassRole`-Berechtigungen, um IAM-Rollen für Aufgaben zu verwenden.

CodeDeploy benötigt Berechtigungen zum Aufrufen von Amazon ECS-APIs, zum Ändern Ihres Elastic Load Balancing, zum Aufrufen von Lambda-Funktionen und zum Beschreiben von CloudWatch Alarmen sowie Berechtigungen, um die von Ihrem Service gewünschte Anzahl in Ihrem Namen zu ändern. Bevor Sie einen Amazon-ECS-Service erstellen, der den Blau/Grün-Bereitstellungstyp verwendet, müssen Sie eine IAM-Rolle erstellen (`ecsCodeDeployRole`). Weitere Informationen finden Sie unter Amazon ECS CodeDeploy IAM-Rolle.

Die Beispiel für einen Amazon ECS-Service erstellen- und Beispiel für einen Amazon ECS-Service aktualisieren-IAM-Richtlinienbeispiele zeigen die Berechtigungen, die für Benutzer erforderlich sind, um Amazon-ECS-Blau/Grün-Bereitstellungen auf der AWS Management Console zu verwenden.

Bereitstellung eines Amazon ECS-Service mithilfe einer blauen/grünen Bereitstellung

Erfahren Sie, wie Sie einen Amazon ECS-Service erstellen, der eine Fargate-Aufgabe enthält, die den Bereitstellungstyp Blau/Grün mit dem verwendet. AWS CLI

> **Note**
>
> Für AWS CloudFormation wurde Unterstützung für die Durchführung einer blau/grünen Bereitstellung hinzugefügt. Weitere Informationen finden Sie im AWS CloudFormation Benutzerhandbuch unter Durchführen von Blue/Green-Bereitstellungen von Amazon ECS CodeDeploy mithilfe von AWS CloudFormation Amazon ECS.

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass Sie die folgenden Voraussetzungen erfüllt haben:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen zur Installation oder Aktualisierung von finden Sie unter Installation von AWS Command Line Interface. AWS CLI
- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben eine VPC und die zu verwendende Sicherheitsgruppe erstellt. Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.
- Die Amazon ECS CodeDeploy IAM-Rolle wird erstellt. Weitere Informationen finden Sie unter Amazon ECS CodeDeploy IAM-Rolle.

Schritt 1: Erstellen eines Application Load Balancers

Amazon-ECS-Dienste, die den Blau/Grün-Bereitstellungstyp verwenden, müssen entweder einen Application Load Balancer oder einen Network Load Balancer verwendet werden. In diesem Tutorial wird ein Application Load Balancer verwendet.

So erstellen Sie einen Application Load Balancer

1. Verwenden des Befehls create-load-balancer, um einen Application Load Balancer zu erstellen. Geben Sie zwei Subnetze, die sich nicht in derselben Availability Zone befinden, sowie eine Sicherheitsgruppe an.

   ```
   aws elbv2 create-load-balancer \
        --name bluegreen-alb \
        --subnets subnet-abcd1234 subnet-abcd5678 \
        --security-groups sg-abcd1234 \
        --region us-east-1
   ```

   Die Ausgabe enthält den Amazon Resource Name (ARN) des Load Balancers im folgenden Format:

   ```
   arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
   e5ba62739c16e642
   ```

2. Verwenden Sie den Befehl create-target-group, um eine Zielgruppe zu erstellen. Diese Zielgruppe leitet den Datenverkehr an den ursprünglichen Aufgabesatz in Ihrem Service weiter.

   ```
   aws elbv2 create-target-group \
        --name bluegreentarget1 \
        --protocol HTTP \
        --port 80 \
        --target-type ip \
        --vpc-id vpc-abcd1234 \
        --region us-east-1
   ```

   Die Ausgabe umfasst den ARN der Zielgruppe im folgenden Format:

   ```
   arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
   bluegreentarget1/209a844cd01825a4
   ```

3. Verwenden Sie den Befehl create-listener, um einen Listener für Ihren Load Balancer mit einer Standardregel zu erstellen, die Anforderungen an Ihre Zielgruppe weiterleitet.

```
aws elbv2 create-listener \
     --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
     --protocol HTTP \
     --port 80 \
     --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup
bluegreentarget1/209a844cd01825a4 \
     --region us-east-1
```

Die Ausgabe enthält den ARN des Listener im folgenden Format:

```
arn:aws:elasticloadbalancing:region:aws_account_id:listener/app/bluegreen-alb/
e5ba62739c16e642/665750bec1b03bd4
```

Schritt 2: Erstellen eines Amazon-ECS-Clusters

Verwenden Sie den Befehl create-cluster, um einen Cluster namens `tutorial-bluegreen-cluster` zu erstellen.

```
aws ecs create-cluster \
     --cluster-name tutorial-bluegreen-cluster \
     --region us-east-1
```

Die Ausgabe enthält den ARN des Clusters im folgenden Format:

```
arn:aws:ecs:region:aws_account_id:cluster/tutorial-bluegreen-cluster
```

Schritt 3: Registrieren einer Aufgabendefinition

Verwenden Sie den Befehl register-task-definition, um eine Aufgabendefinition zu registrieren, die mit Fargate kompatibel ist. Für sie muss als Netzwerkmodus `awsvpc` verwendet werden. Nachfolgend finden Sie die Beispiel-Aufgabendefinition, die für dieses Tutorial verwendet wurde.

Erstellen Sie zuerst eine Datei mit dem Namen `fargate-task.json` und dem folgenden Inhalt. Stellen Sie sicher, dass Sie den ARN für die Aufgabenausführungsrolle verwenden. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "httpd:2.4",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #00FFFF;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512",
        "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole"
}
```

Registrieren Sie dann die Aufgabendefinition mit der von Ihnen erstellten Datei `fargate-task.json`.

```
aws ecs register-task-definition \
     --cli-input-json file://fargate-task.json \
     --region us-east-1
```

Schritt 4: Erstellen Sie einen Amazon-ECS-Service

Verwenden Sie den Befehl create-service, um einen Service zu erstellen.

Erstellen Sie zuerst eine Datei mit dem Namen `service-bluegreen.json` und dem folgenden Inhalt.

```
{
    "cluster": "tutorial-bluegreen-cluster",
    "serviceName": "service-bluegreen",
    "taskDefinition": "tutorial-task-def",
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4",
            "containerName": "sample-app",
            "containerPort": 80
        }
    ],
    "launchType": "FARGATE",
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "CODE_DEPLOY"
    },
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234", "subnet-abcd5678" ]
       }
    },
    "desiredCount": 1
}
```

Erstellen Sie dann Ihren Service unter Verwendung der Datei `service-bluegreen.json`, die Sie erstellt haben.

```
aws ecs create-service \
     --cli-input-json file://service-bluegreen.json \
     --region us-east-1
```

Die Ausgabe enthält den ARN des Service im folgenden Format:

```
arn:aws:ecs:region:aws_account_id:service/service-bluegreen
```

Rufen Sie den DNS-Namen des Load Balancer mit dem folgenden Befehl ab.

```
 aws elbv2 describe-load-balancers --name bluegreen-alb  --query
 'LoadBalancers[*].DNSName'
```

Geben Sie den DNS-Namen in Ihren Webbrowser ein. Sie sollten eine Webseite sehen, die die Beispielanwendung mit einem blauen Hintergrund anzeigt.

Schritt 5: Erstellen der AWS CodeDeploy -Ressourcen

Gehen Sie wie folgt vor, um Ihre CodeDeploy Anwendung, die Application Load Balancer Balancer-Zielgruppe für die CodeDeploy Bereitstellungsgruppe und die CodeDeploy Bereitstellungsgruppe zu erstellen.

Um Ressourcen zu erstellen CodeDeploy

1. Verwenden Sie den Befehl create-application, um eine CodeDeploy Anwendung zu erstellen. Geben Sie die ECS-Plattform für die Datenverarbeitung an.

   ```
   aws deploy create-application \
        --application-name tutorial-bluegreen-app \
        --compute-platform ECS \
        --region us-east-1
   ```

   Die Ausgabe enthält die Anwendungs-ID im folgenden Format:

   ```
   {
       "applicationId": "b8e9c1ef-3048-424e-9174-885d7dc9dc11"
   }
   ```

2. Verwenden Sie den Befehl create-target-group, um eine zweite Application Load Balancer Balancer-Zielgruppe zu erstellen, die bei der Erstellung Ihrer Bereitstellungsgruppe verwendet wird. CodeDeploy

```
aws elbv2 create-target-group \
     --name bluegreentarget2 \
     --protocol HTTP \
     --port 80 \
     --target-type ip \
     --vpc-id "vpc-0b6dd82c67d8012a1" \
     --region us-east-1
```

Die Ausgabe umfasst den ARN der Zielgruppe im folgenden Format:

```
arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc
```

3. Verwenden Sie den Befehl create-deployment-group, um eine Bereitstellungsgruppe zu erstellen. CodeDeploy

Erstellen Sie zuerst eine Datei mit dem Namen `tutorial-deployment-group.json` und dem folgenden Inhalt. Dieses Beispiel verwendet die von Ihnen erstellte Ressource. Geben Sie für die `serviceRoleArn` den ARN Ihrer Amazon CodeDeploy ECS-IAM-Rolle an. Weitere Informationen finden Sie unter Amazon ECS CodeDeploy IAM-Rolle.

```
{
   "applicationName": "tutorial-bluegreen-app",
   "autoRollbackConfiguration": {
      "enabled": true,
      "events": [ "DEPLOYMENT_FAILURE" ]
   },
   "blueGreenDeploymentConfiguration": {
      "deploymentReadyOption": {
         "actionOnTimeout": "CONTINUE_DEPLOYMENT",
         "waitTimeInMinutes": 0
      },
      "terminateBlueInstancesOnDeploymentSuccess": {
         "action": "TERMINATE",
         "terminationWaitTimeInMinutes": 5
      }
   },
```

```
   "deploymentGroupName": "tutorial-bluegreen-dg",
   "deploymentStyle": {
      "deploymentOption": "WITH_TRAFFIC_CONTROL",
      "deploymentType": "BLUE_GREEN"
   },
   "loadBalancerInfo": {
      "targetGroupPairInfoList": [
        {
          "targetGroups": [
             {
                 "name": "bluegreentarget1"
             },
             {
                 "name": "bluegreentarget2"
             }
          ],
          "prodTrafficRoute": {
              "listenerArns": [
                  "arn:aws:elasticloadbalancing:region:aws_account_id:listener/
app/bluegreen-alb/e5ba62739c16e642/665750bec1b03bd4"
              ]
          }
        }
      ]
   },
   "serviceRoleArn": "arn:aws:iam::aws_account_id:role/ecsCodeDeployRole",
   "ecsServices": [
       {
           "serviceName": "service-bluegreen",
           "clusterName": "tutorial-bluegreen-cluster"
       }
   ]
}
```

Erstellen Sie dann die CodeDeploy Bereitstellungsgruppe.

```
aws deploy create-deployment-group \
     --cli-input-json file://tutorial-deployment-group.json \
     --region us-east-1
```

Die Ausgabe enthält die Bereitstellungsgruppen-ID im folgenden Format:

```
{
    "deploymentGroupId": "6fd9bdc6-dc51-4af5-ba5a-0a4a72431c88"
}
```

Schritt 6: Erstellen und überwachen Sie eine CodeDeploy Bereitstellung

Bevor Sie eine CodeDeploy Bereitstellung erstellen, aktualisieren Sie die Aufgabendefinition `fargate-task.json` wie folgt, um die Hintergrundfarbe der Beispiel-App `command` in Grün zu ändern.

```
{
    ...
        "containerDefinitions": [
            {
                ...
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #097969;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
                ]
            }
        ],
       ...
}
```

Registrieren Sie die aktualisierte Aufgabendefinition mit dem folgenden Befehl.

```
aws ecs register-task-definition \
     --cli-input-json file://fargate-task.json \
     --region us-east-1
```

Gehen Sie nun wie folgt vor, um eine Anwendungsspezifikationsdatei (AppSpec Datei) und eine CodeDeploy Bereitstellung zu erstellen und hochzuladen.

Um eine CodeDeploy Bereitstellung zu erstellen und zu überwachen

1. Gehen Sie wie folgt vor, um eine AppSpec Datei zu erstellen und hochzuladen.

a. Erstellen Sie eine Datei `appspec.yaml` mit dem Namen des Inhalts der CodeDeploy Bereitstellungsgruppe. In diesem Beispiel wird die aktualisierte Aufgabendefinition verwendet.

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "arn:aws:ecs:region:aws_account_id:task-
definition/tutorial-task-def:2"
        LoadBalancerInfo:
          ContainerName: "sample-app"
          ContainerPort: 80
        PlatformVersion: "LATEST"
```

b. Verwenden Sie den Befehl s3 mb, um einen Amazon S3 S3-Bucket für die AppSpec Datei zu erstellen.

```
aws s3 mb s3://tutorial-bluegreen-bucket
```

c. Verwenden Sie den Befehl s3 cp, um die AppSpec Datei in den Amazon S3 S3-Bucket hochzuladen.

```
aws s3 cp ./appspec.yaml s3://tutorial-bluegreen-bucket/appspec.yaml
```

2. Erstellen Sie die CodeDeploy Bereitstellung mithilfe der folgenden Schritte.

a. Erstellen Sie eine Datei `create-deployment.json` mit dem Namen des Inhalts der CodeDeploy Bereitstellung. Dieses Beispiel verwendet die Ressourcen, die Sie zuvor im Tutorial erstellt haben.

```
{
    "applicationName": "tutorial-bluegreen-app",
    "deploymentGroupName": "tutorial-bluegreen-dg",
    "revision": {
        "revisionType": "S3",
        "s3Location": {
            "bucket": "tutorial-bluegreen-bucket",
            "key": "appspec.yaml",
            "bundleType": "YAML"
```

```
        }
    }
}
```

b. Verwenden Sie den Befehl create-deployment zum Erstellen der Bereitstellung.

```
aws deploy create-deployment \
     --cli-input-json file://create-deployment.json \
     --region us-east-1
```

Die Ausgabe enthält die Bereitstellungs-ID im folgenden Format:

```
{
    "deploymentId": "d-RPCR1U3TW"
}
```

3. Verwenden Sie den Befehl get-deployment-target, um die Details der Bereitstellung unter Angabe der `deploymentId` aus der vorherigen Ausgabe anzufordern.

```
aws deploy get-deployment-target \
--deployment-id "d-IMJU3A8TW" \
--target-id tutorial-bluegreen-cluster:service-bluegreen \
--region us-east-1
```

Anfänglich lautet der Bereitstellungsstatus `InProgress`. Der Datenverkehr wird an den ursprünglichen Aufgabensatz weitergeleitet, der ein `taskSetLabel` von `BLUE` hat, einen Status von `PRIMARY` und ein `trafficWeight` von `100.0`. Der Ersatz-Tasksatz hat ein `taskSetLabel` von `GREEN`, eine Status von `ACTIVE` und ein `trafficWeight` von `0.0`. Der Webbrowser, in den Sie den DNS-Namen eingegeben haben, zeigt die Beispiel-App weiterhin mit blauem Hintergrund an.

```
{
"deploymentTarget": {
"deploymentTargetType": "ECSTarget",
"ecsTarget": {
    "deploymentId": "d-RPCR1U3TW",
    "targetId": "tutorial-bluegreen-cluster:service-bluegreen",
    "targetArn": "arn:aws:ecs:region:aws_account_id:service/service-bluegreen",
    "lastUpdatedAt": "2023-08-10T12:07:24.797000-05:00",
    "lifecycleEvents": [
        {
```

```
            "lifecycleEventName": "BeforeInstall",
            "startTime": "2023-08-10T12:06:22.493000-05:00",
            "endTime": "2023-08-10T12:06:22.790000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "Install",
            "startTime": "2023-08-10T12:06:22.936000-05:00",
            "status": "InProgress"
        },
        {
            "lifecycleEventName": "AfterInstall",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "BeforeAllowTraffic",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "AllowTraffic",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "AfterAllowTraffic",
            "status": "Pending"
        }
    ],
    "status": "InProgress",
    "taskSetsInfo": [
        {
            "identifer": "ecs-svc/9223370493423413672",
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
            "status": "ACTIVE",
            "trafficWeight": 0.0,
            "targetGroup": {
                "name": "bluegreentarget2"
            },
            "taskSetLabel": "Green"
        },
        {
            "identifer": "ecs-svc/9223370493425779968",
            "desiredCount": 1,
```

```
            "pendingCount": 0,
            "runningCount": 1,
            "status": "PRIMARY",
            "trafficWeight": 100.0,
            "targetGroup": {
                "name": "bluegreentarget1"
            },
            "taskSetLabel": "Blue"
        }
    ]
}
}
}
```

Fahren Sie mit dem Abrufen der Bereitstellungsdetails mithilfe des Befehls fort, bis der Bereitstellungsstatus `Succeeded` lautet, wie in der folgenden Ausgabe gezeigt. Der Datenverkehr wird nun zum Ersatz-Aufgabensatz umgeleitet, der nun einen Status von `PRIMARY` und einen `trafficWeight` von `100.0` hat. Aktualisieren Sie den Webbrowser, in den Sie den DNS-Namen des Load Balancer eingegeben haben, und die Beispielanwendung sollte nun mit einem grünen Hintergrund angezeigt werden.

```
{
"deploymentTarget": {
"deploymentTargetType": "ECSTarget",
"ecsTarget": {
    "deploymentId": "d-RPCR1U3TW",
    "targetId": "tutorial-bluegreen-cluster:service-bluegreen",
    "targetArn": "arn:aws:ecs:region:aws_account_id:service/service-bluegreen",
    "lastUpdatedAt": "2023-08-10T12:07:24.797000-05:00",
    "lifecycleEvents": [
        {
            "lifecycleEventName": "BeforeInstall",
            "startTime": "2023-08-10T12:06:22.493000-05:00",
            "endTime": "2023-08-10T12:06:22.790000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "Install",
            "startTime": "2023-08-10T12:06:22.936000-05:00",
            "endTime": "2023-08-10T12:08:25.939000-05:00",
            "status": "Succeeded"
        },
```

```
        {
            "lifecycleEventName": "AfterInstall",
            "startTime": "2023-08-10T12:08:26.089000-05:00",
            "endTime":  "2023-08-10T12:08:26.403000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "BeforeAllowTraffic",
            "startTime": "2023-08-10T12:08:26.926000-05:00",
            "endTime":  "2023-08-10T12:08:27.256000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "AllowTraffic",
            "startTime": "2023-08-10T12:08:27.416000-05:00",
            "endTime": "2023-08-10T12:08:28.195000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "AfterAllowTraffic",
            "startTime": "2023-08-10T12:08:28.715000-05:00",
            "endTime":  "2023-08-10T12:08:28.994000-05:00",
            "status": "Succeeded"
        }
    ],
    "status": "Succeeded",
    "taskSetsInfo": [
        {
            "identifer": "ecs-svc/9223370493425779968",
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
            "status": "ACTIVE",
            "trafficWeight": 0.0,
            "targetGroup": {
                "name": "bluegreentarget1"
            },
            "taskSetLabel": "Blue"
        },
        {
            "identifer": "ecs-svc/9223370493423413672",
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
```

```
            "status": "PRIMARY",
            "trafficWeight": 100.0,
            "targetGroup": {
                "name": "bluegreentarget2"
            },
            "taskSetLabel": "Green"
        }
    ]
}
}
}
```

Schritt 7: Bereinigen

Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie die damit verknüpften Ressourcen bereinigen, um zu vermeiden, dass für nicht verwendete Ressourcen Kosten entstehen.

Bereinigen der Ressourcen des Tutorials

1. Verwenden Sie den Befehl delete-deployment-group, um die Bereitstellungsgruppe zu löschen. CodeDeploy

```
aws deploy delete-deployment-group \
     --application-name tutorial-bluegreen-app \
     --deployment-group-name tutorial-bluegreen-dg \
     --region us-east-1
```

2. Verwenden Sie den Befehl delete-application, um die Anwendung zu löschen. CodeDeploy

```
aws deploy delete-application \
     --application-name tutorial-bluegreen-app \
     --region us-east-1
```

3. Verwenden Sie den Befehl delete-service, um den Amazon-ECS-Service zu löschen. Mit dem `--force`-Flag können Sie einen Service auch dann löschen, wenn er nicht auf null Aufgaben herunterskaliert wurde.

```
aws ecs delete-service \
     --service arn:aws:ecs:region:aws_account_id:service/service-bluegreen \
     --force \
     --region us-east-1
```

4. Verwenden Sie den Befehl delete-cluster, um den Amazon-ECS-Cluster zu löschen.

```
aws ecs delete-cluster \
     --cluster tutorial-bluegreen-cluster \
     --region us-east-1
```

5. Verwenden Sie den Befehl s3 rm, um die AppSpec Datei aus dem Amazon S3 S3-Bucket zu löschen.

```
aws s3 rm s3://tutorial-bluegreen-bucket/appspec.yaml
```

6. Verwenden Sie den Befehl s3 rb zum Löschen des Amazon S3-Buckets.

```
aws s3 rb s3://tutorial-bluegreen-bucket
```

7. Um Ihren Application Load Balancer zu löschen, verwenden Sie den Befehl delete-load-balancer.

```
aws elbv2 delete-load-balancer \
     --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
     --region us-east-1
```

8. Verwenden Sie den Befehl delete-target-group, um die zwei Application Load Balancer-Zielgruppen zu löschen.

```
aws elbv2 delete-target-group \
     --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4 \
     --region us-east-1
```

```
aws elbv2 delete-target-group \
     --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc \
     --region us-east-1
```

Stellen Sie Amazon ECS-Services mithilfe eines Controllers eines Drittanbieters bereit

Der external (externe) Bereitstellungstyp erlaubt Ihnen, einen beliebigen Bereitstellungscontroller eines Drittanbieters zu verwenden, um die volle Kontrolle über den Bereitstellungsprozess für einen Amazon-ECS-Service zu haben. Die Details für Ihren Service werden entweder mit den API-Aktionen der Serviceverwaltung (`CreateService`, `UpdateService` und `DeleteService`) oder den API-Aktionen der Aufgabensatzverwaltung (`CreateTaskSet`, `UpdateTaskSet`, `UpdateServicePrimaryTaskSet` und `DeleteTaskSet`) verwaltet. Jede API-Aktion verwaltet eine Teilmenge der Servicedefinitions-Parameter.

Die `UpdateService`-API-Aktion aktualisiert die Parameter für die gewünschte Anzahl und die Übergangsfrist der Zustandsprüfung für einen Service. Wenn der Starttyp oder die Plattformversion, Load Balancer-Details, Netzwerkkonfiguration oder Aufgabendefinition aktualisiert werden müssen, müssen Sie einen neuen Aufgabesatz erstellen.

Die `UpdateTaskSet`-API-Aktion aktualisiert nur den Scale-Parameter für einen Aufgabensatz.

Die `UpdateServicePrimaryTaskSet`-API-Aktion ändert, welcher Aufgabensatz in einem Service als primärer Aufgabensatz fungiert. Wenn Sie die API-Aktion `DescribeServices` aufrufen, gibt sie alle für den primären Aufgabensatz angegebenen Felder zurück. Wenn der primäre Aufgabensatz eines Service aktualisiert wird, werden alle Parameterwerte des neuen primären Aufgabensatzes, die von denen des alten primären Aufgabensatzes in einem Service abweichen, beim Definieren des neuen primären Aufgabensatzes auf den neuen Wert aktualisiert. Wenn für einen Service kein primärer Aufgabensatz definiert ist, sind die Werte der Aufgabensatzfelder beim Beschreiben des Service null.

Überlegungen zur externen Bereitstellung

Beachten Sie Folgendes, wenn Sie den externen Bereitstellungstyp verwenden:

- Die unterstützten Load Balancer-Typen sind entweder ein Application Load Balancer oder ein Network Load Balancer.
- Aufgaben, die den Starttyp Fargate oder die Controller-Typen `EXTERNAL` verwenden, unterstützen die `DAEMON`-Planungsstrategie nicht.

Workflow für externe Bereitstellung

Im Folgenden finden Sie den grundlegenden Arbeitsablauf für die Verwaltung einer externen Bereitstellung auf Amazon ECS.

So verwalten Sie einen Amazon-ECS-Service mithilfe eines externen Bereitstellungs-Controllers

1. Erstellen Sie einen Amazon-ECS-Service. Der einzige erforderliche Parameter ist der Servicename. Wenn Sie einen Service mit einem externen Bereitstellungs-Controller erstellen, können Sie die folgenden Parameter angeben. Alle anderen Service-Parameter werden bei der Erstellung eines Aufgabesatzes innerhalb des Service festlegt.

   `serviceName`

   Typ: Zeichenfolge

   Erforderlich: Ja

   Der Name Ihres Service. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig. Servicenamen in einem Cluster müssen eindeutig sein. Sie können jedoch ähnlich benannte Services in mehreren Clustern innerhalb einer Region oder in mehreren Regionen haben.

   `desiredCount`

   Die Anzahl der Instanziierungen der angegebenen Aufgabensatzdefinition, die innerhalb des Service platziert und ausgeführt werden sollen.

   `deploymentConfiguration`

   Optionale Bereitstellungsparameter zur Steuerung, wie viele Aufgaben während einer Bereitstellung ausgeführt werden, und zur Steuerung der Reihenfolge beim Starten oder Stoppen von Aufgaben.

   `tags`

   Typ: Array von -Objekten

   Erforderlich: Nein

   Die Metadaten, die Sie auf den Service anwenden, um die Kategorisierung und Organisation zu erleichtern. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen. Wenn ein Service gelöscht wird, werden auch die

Tags gelöscht. Es können maximal 50 Tags auf den Service angewendet werden. Weitere Informationen finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.

`key`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 128 Zeichen.

Erforderlich: Nein

Ein Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Schlüssel ist eine allgemeine Bezeichnung, die wie eine Kategorie für spezifischere Tag-Werte fungiert.

`value`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

Der optionale Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Wert fungiert als Deskriptor in einer Tag-Kategorie (Schlüssel).

`enableECSManagedTags`

Gibt an, ob von Amazon ECS verwaltete Tags für Aufgaben im Service verwendet werden sollen. Weitere Informationen finden Sie unter Verwenden Sie Tags für die Abrechnung.

`propagateTags`

Typ: Zeichenfolge

Zulässige Werte: `TASK_DEFINITION` | `SERVICE`

Erforderlich: Nein

Gibt an, ob die Tags von der Aufgabendefinition oder dem Service in die Aufgaben in dem Service kopiert werden sollen. Wenn kein Wert angegeben wird, werden die Tags nicht kopiert. Tags können nur während der Serviceerstellung in die Aufgaben in dem Service kopiert werden. Wenn Sie Tags nach der Service- oder Aufgabenerstellung einer Aufgabe hinzufügen möchten, verwenden Sie die API-Aktion `TagResource`.

`schedulingStrategy`

Die Einplanungsstrategie, die verwendet werden soll. Services mit einem externen Deployment-Controller unterstützen nur die Planungsstrategie `REPLICA`.

`placementConstraints`

Ein Array von Platzierungseinschränkungsobjekten, die für Aufgaben in Ihrem Service verwendet werden sollen. Sie können maximal zehn Einschränkungen pro Aufgabe festlegen (dieses Limit enthält Einschränkungen in der Aufgabendefinition und solche, die während der Laufzeit festgelegt werden). Wenn Sie den Starttyp Fargate verwenden, werden keine Platzierungsbedingungen für die Aufgaben unterstützt.

`placementStrategy`

Die Platzierungsstrategieobjekte, die für Aufgaben in Ihrem Service verwendet werden sollen. Sie können maximal vier Strategieregeln pro Service festlegen.

Im Folgenden finden Sie ein Beispiel für eine Servicedefinition, mit der ein Service über einen externen Bereitstellungs-Controller erstellt wird.

```
{
    "cluster": "",
    "serviceName": "",
    "desiredCount": 0,
    "role": "",
    "deploymentConfiguration": {
        "maximumPercent": 0,
        "minimumHealthyPercent": 0
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
```

```
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION"
}
```

2. Erstellen Sie einen anfänglichen Aufgabensatz. Der Aufgabesatz enthält die folgenden Details zu Ihrem Service:

   `taskDefinition`

   Die zu verwendende Aufgabendefinition für die Aufgaben im Aufgabensatz.

   `launchType`

   Typ: Zeichenfolge

   Zulässige Werte: `EC2` | `FARGATE` | `EXTERNAL`

   Erforderlich: Nein

   Der Starttyp, der für die Ausführung Ihres Service verwendet wird. Ist kein Starttyp angegeben, wird standardmäßig `capacityProviderStrategy` verwendet. Weitere Informationen finden Sie unter Amazon-ECS-Starttypen.

   Wenn eine `launchType` angegeben ist, muss der `capacityProviderStrategy`-Parameter weggelassen werden.

   `platformVersion`

   Typ: Zeichenfolge

   Erforderlich: Nein

Die Plattformversion, auf der Ihre Aufgaben im Service ausgeführt werden. Eine Plattformversion ist nur für Aufgaben mit dem Starttyp Fargate vorgesehen. Ist kein solcher angegeben, wird standardmäßig die neueste Version (`LATEST`) verwendet.

AWS Fargate-Plattformversionen werden verwendet, um auf eine bestimmte Laufzeitumgebung für die Fargate-Task-Infrastruktur zu verweisen. Bei der Angabe der `LATEST`-Plattformversion bei Ausführung einer Aufgabe oder beim Erstellen eines Service erhalten Sie die aktuellste Plattformversion, die für Ihre Aufgaben zur Verfügung steht. Wenn Sie Ihren Service skalieren, erhalten diese Aufgaben die Plattformversion, die in der aktuellen Bereitstellung des Service angegeben wurde. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

> **Note**
>
> Plattformversionen werden nicht für Aufgaben angegeben, die den Starttyp EC2 verwenden.

`loadBalancers`

Ein Load Balancer-Objekt, das den Load Balancer angibt, der mit Ihrem Service verwendet werden soll. Wenn Sie einen externen Bereitstellungscontroller verwenden, werden nur Application Load Balancer und Network Load Balancer unterstützt. Wenn Sie einen Application Load Balancer verwenden, ist pro Aufgabensatz nur eine Application Load Balancer-Zielgruppe zulässig.

Der folgende Ausschnitt zeigt ein `loadBalancer`-Beispielobjekt.

```
"loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
]
```

> **Note**
>
> Wenn Sie ein `loadBalancer`-Objekt angeben, müssen Sie einen `targetGroupArn` angeben und die Parameter `loadBalancerName` auslassen.

`networkConfiguration`

Die Netzwerkkonfiguration für den Service. Dieser Parameter ist erforderlich, damit Aufgabendefinitionen, die den Netzwerkmodus `awsvpc` verwenden, ihre eigene Elastic-Network-Schnittstelle erhalten. Für andere Netzwerkmodi wird er nicht unterstützt. Weitere Informationen über Netzwerke für den Fargate-Starttyp finden Sie unterNetzwerkoptionen für Amazon ECS-Aufgaben für den Starttyp Fargate.

`serviceRegistries`

Die Details der Service Discovery-Registrierungen, die diesem Service zugewiesen werden sollen. Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.

`scale`

Ein Gleitkomma-Prozentsatz der gewünschten Anzahl von Aufgaben, die im Aufgabensatz platziert und ausgeführt werden sollen. Der Wert wird als Gesamtprozentsatz des `desiredCount`-Wertes eines Services angegeben. Akzeptierte Werte sind Zahlen zwischen 0 und 100.

Im Folgenden finden Sie ein JSON-Beispiel für die Erstellung eines Aufgabensatzes für einen externen Bereitstellungs-Controller.

```
{
    "service": "",
    "cluster": "",
    "externalId": "",
    "taskDefinition": "",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
```

```
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "scale": {
        "value": null,
        "unit": "PERCENT"
    },
    "clientToken": ""
}
```

3. Wenn Service-Änderungen erforderlich sind, verwenden Sie abhängig von den zu aktualisierenden Parametern die API-Aktion `UpdateService`, `UpdateTaskSet` oder `CreateTaskSet`. Wenn Sie eine Aufgabe erstellt haben, bestimmen Sie mithilfe des Parameters `scale` für jede Aufgabe in einem Service, wie viele Aufgaben im Service weiterhin ausgeführt werden sollen. Wenn ein Service beispielsweise `tasksetA` enthält und Sie `tasksetB` erstellen, können Sie zuerst die Gültigkeit von `tasksetB` testen, bevor Sie den

Produktionsdatenverkehr darauf umstellen. Sie können als `scale`-Wert für beide Aufgabensätze `100` einstellen. Wenn Sie dann bereit sind, den gesamten Produktionsdatenverkehr auf `tasksetB` umzustellen, können Sie den `scale`-Wert für `tasksetA` auf `0` aktualisieren, um ihn nach unten zu skalieren.

Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen

Ihr Service kann optional so konfiguriert werden, dass er Elastic Load Balancing verwendet, um den Traffic gleichmäßig auf die Aufgaben in Ihrem Service zu verteilen.

> **Note**
>
> Wenn Sie Aufgabensätze verwenden, müssen alle Aufgaben im Satz so konfiguriert sein, dass sie entweder Elastic Load Balancing verwenden oder Elastic Load Balancing nicht verwenden.

Amazon ECS-Services, die auf gehostet werden, AWS Fargate unterstützen die Application Load Balancers, Network Load Balancers und Gateway Load Balancers. Anhand der folgenden Tabelle erfahren Sie, welche Art von Load Balancer Sie verwenden sollten.

Load Balancer Balancer-Typ	In diesen Fällen verwenden	
Application Load Balancer	Leiten Sie den HTTP/HTTPS-Verkehr (oder Layer 7) weiter. Application Load Balancers bieten verschiedene Funktionen an, die sie besonders attraktiv für die Verwendung mit Amazon-ECS-Services machen: • Jeder Service kann den Datenverkehr von mehreren Load Balancern abwickeln	

Load Balancer Balancer-Typ	In diesen Fällen verwenden	
	und mehrere Ports mit Lastenausgleich verfügbar machen, indem er mehrere Zielgruppen festlegt. • Sie werden von Aufgaben unterstützt, die auf Fargate und EC2-Instances gehostet werden. • Application Load Balancers ermöglichen es, dass Container die dynamisch e Host-Port-Zuweisun g verwenden (so dass mehrere Aufgaben von demselben Service pro Container-Instance erlaubt sind). • Application Load Balancers unterstützen pfadbasie rte Routing- und Priorität sregeln (sodass mehrere Services denselben Listener-Port auf einem einzelnen Application Load Balancer verwenden können).	
Network Load Balancer	Leiten Sie TCP- oder UDP-Verkehr (oder Layer-4-V erkehr) weiter.	

Load Balancer Balancer-Typ	In diesen Fällen verwenden	
Gateway Load Balancer	Leiten Sie TCP- oder UDP-Verkehr (oder Layer-4-V erkehr) weiter. Verwenden Sie virtuelle Appliances wie Firewalls, Systeme zur Erkennung und Verhinderung von Eindringl ingen und Deep-Packet-Inspektionssysteme.	

Wir empfehlen Ihnen, Application Load Balancers für Ihre Amazon ECS-Services zu verwenden, damit Sie diese neuesten Funktionen nutzen können, es sei denn, Ihr Service erfordert eine Funktion, die nur mit Network Load Balancers oder Gateway Load Balancers verfügbar ist. Weitere Informationen über Elastic Load Balancing und die Unterschiede zwischen den Load Balancer-Typen finden Sie unter Benutzerhandbuch für Elastic Load Balancing.

Mit Ihrem Load Balancer zahlen Sie nur für das, was Sie auch tatsächlich nutzen. Weitere Informationen finden Sie unter Elastic Load Balancing Pricing.

Optimieren Sie die Parameter für die Zustandsprüfung des Load Balancers für Amazon ECS

Load Balancer leiten Anfragen nur an die fehlerfreien Ziele in den Availability Zones für den Load Balancer weiter. Jedes Ziel ist für eine Zielgruppe registriert. Der Load Balancer überprüft den Zustand jedes Ziels anhand der Einstellungen für die Zustandsprüfung der Zielgruppe. Nachdem Sie das Ziel registriert haben, muss es eine Integritätsprüfung bestehen, um als fehlerfrei eingestuft zu werden. Amazon ECS überwacht den Load Balancer. Der Load Balancer sendet regelmäßig Integritätsprüfungen an den Amazon ECS-Container. Der Amazon ECS-Agent überwacht den Zustand des Containers und wartet darauf, dass der Load Balancer Bericht erstattet. Dies geschieht, bevor der Container als fehlerfrei eingestuft wird.

Zwei Parameter für die Integritätsprüfung von Elastic Load Balancing wirken sich auf die Bereitstellungsgeschwindigkeit aus:

- Intervall Health Integritätsprüfungen: Bestimmt den ungefähren Zeitraum in Sekunden zwischen den Zustandsprüfungen eines einzelnen Containers. Standardmäßig überprüft der Load Balancer alle 30 Sekunden.

  Dieser Parameter heißt:

  - `HealthCheckIntervalSeconds`in der Elastic Load Balancing API
  - Intervall auf der Amazon EC2 EC2-Konsole
- Anzahl fehlerhafter Schwellenwerte: Bestimmt die Anzahl der aufeinanderfolgenden erfolgreichen Zustandsprüfungen, die erforderlich sind, bevor ein fehlerhafter Container als fehlerfrei eingestuft wird. Standardmäßig benötigt der Load Balancer fünf bestandene Zustandsprüfungen, bevor er meldet, dass der Zielcontainer fehlerfrei ist.

  Dieser Parameter trägt den Namen:

  - `HealthyThresholdCount`in der Elastic Load Balancing API
  - Fehlerfreier Schwellenwert auf der Amazon EC2 EC2-Konsole

Bei den Standardeinstellungen beträgt die Gesamtzeit zur Bestimmung des Zustands eines Containers zwei Minuten und 30 Sekunden (`30 seconds * 5 = 150 seconds`).

Sie können den Integritätsprüfungsprozess beschleunigen, wenn Ihr Dienst in weniger als 10 Sekunden gestartet und stabilisiert wird. Um den Vorgang zu beschleunigen, reduzieren Sie die Anzahl der Integritätsprüfungen und das Intervall zwischen den Prüfungen.

- `HealthCheckIntervalSeconds`(Elastic Load Balancing API-Name) oder Interval (Name der Amazon EC2 EC2-Konsole): 5
- `HealthyThresholdCount`(Elastic Load Balancing API-Name) oder Health-Schwellenwert (Name der Amazon EC2 EC2-Konsole): 2

Mit dieser Einstellung dauert die Integritätsprüfung 10 Sekunden im Vergleich zur Standardeinstellung von zwei Minuten und 30 Sekunden.

Weitere Informationen zu den Elastic Load Balancing Health Check-Parametern finden Sie unter Health Checks für Ihre Zielgruppen im Elastic Load Balancing User Guide.

Optimieren Sie die Load Balancer-Verbindungsparameter für Amazon ECS

Um eine Optimierung zu ermöglichen, halten die Clients eine permanente Verbindung zum Container-Service aufrecht. Auf diese Weise können nachfolgende Anfragen von diesem Client die bestehende Verbindung wiederverwenden. Wenn Sie den Verkehr zu einem Container unterbrechen möchten, benachrichtigen Sie den Load Balancer. Der Load Balancer überprüft regelmäßig, ob der Client die Keep-Alive-Verbindung geschlossen hat. Der Amazon ECS-Agent überwacht den Load Balancer und wartet darauf, dass der Load Balancer meldet, dass die Keep-Alive-Verbindung geschlossen ist (das Ziel befindet sich in einem `UNUSED` Status).

Die Zeitspanne, die der Load Balancer darauf wartet, das Ziel in den `UNUSED` Status zu versetzen, entspricht der Verzögerung bei der Abmeldung. Sie können den folgenden Load Balancer-Parameter konfigurieren, um Ihre Bereitstellungen zu beschleunigen.

- `deregistration_delay.timeout_seconds`: 300 (Standard)

Wenn Sie einen Service mit einer Antwortzeit unter einer Sekunde haben, setzen Sie den Parameter auf den folgenden Wert, damit der Load Balancer nur 5 Sekunden wartet, bevor er die Verbindung zwischen dem Client und dem Back-End-Service unterbricht:

- `deregistration_delay.timeout_seconds`: 5

> **Note**
>
> Setzen Sie den Wert nicht auf 5 Sekunden, wenn Sie einen Dienst mit lang anhaltenden Anfragen haben, wie z. B. langsame Datei-Uploads oder Streaming-Verbindungen.

Reaktionsfähigkeit von SIGTERM

Amazon ECS sendet zunächst ein SIGTERM-Signal an die Aufgabe, um zu benachrichtigen, dass die Anwendung beendet und heruntergefahren werden muss. Dann sendet Amazon ECS eine SIGKILL-Nachricht. Wenn Anwendungen den SIGTERM ignorieren, muss der Amazon ECS-Service warten, bis er das SIGKILL-Signal sendet, um den Prozess zu beenden.

Wie lange Amazon ECS auf das Senden der SIGKILL-Nachricht wartet, wird durch die folgende Amazon ECS-Agentenoption bestimmt:

- `ECS_CONTAINER_STOP_TIMEOUT`: 30 (Standard)

Weitere Informationen zum Container-Agent-Parameter finden Sie unter Amazon ECS Container Agent on GitHub.

Um die Wartezeit zu verkürzen, setzen Sie den Amazon ECS-Agentenparameter auf den folgenden Wert:

> **Note**
>
> Wenn Ihre Anwendung länger als 1 Sekunde benötigt, multiplizieren Sie den Wert mit 2 und verwenden Sie diese Zahl als Wert.

- `ECS_CONTAINER_STOP_TIMEOUT`: 2

In diesem Fall wartet Amazon ECS 2 Sekunden, bis der Container heruntergefahren wird, und Amazon ECS sendet dann eine SIGKILL-Nachricht, wenn die Anwendung nicht gestoppt wurde.

Sie können den Anwendungscode auch ändern, um das SIGTERM-Signal abzufangen und darauf zu reagieren. Das Folgende ist ein Beispiel in JavaScript:

```
process.on('SIGTERM', function() {
  server.close();
})
```

Dieser Code bewirkt, dass der HTTP-Server nicht mehr auf neue Anfragen wartet, alle laufenden Anfragen beantwortet und dann der Prozess Node.js beendet wird. Das liegt daran, dass die zugehörige Ereignisschleife nichts mehr zu tun hat. Aus diesem Grund wird der Prozess vorzeitig beendet, wenn er nur 500 ms benötigt, um seine laufenden Anfragen zu beenden, ohne dass der Stop-Timeout abgewartet und ein SIGKILL gesendet werden muss.

Verwenden Sie einen Application Load Balancer für Amazon ECS

Ein Application Load Balancer trifft Routing-Entscheidungen auf Anwendungsebene (HTTP/HTTPS), unterstützt pfadbasiertes Routing und kann Anfragen an einen oder mehrere Ports jeder Container-Instance in Ihrem Cluster weiterleiten. Application Load Balancer unterstützen dynamische Host-Port-Zuweisung. Wenn die Containerdefinition Ihrer Aufgabe beispielsweise Port 80 als NGINX-Container-Port and Port 0 als Host-Port angibt, wird der Host-Port dynamisch aus dem flüchtigen Port-Bereich der Container-Instance ausgewählt (z. B. 32768 bis 61000 beim aktuellen Amazon-

ECS-optimierten AMI). Wenn die Aufgabe gestartet wird, wird der NGINX-Container beim Application Load Balancer als Kombination aus Instanz-ID und Port registriert, und der Datenverkehr wird an die Instance-ID und den Port verteilt, die diesem Container entsprechen. Aufgrund dieser dynamischen Zuordnung können Sie mehrere Aufgaben über einen einzelnen Service auf derselben Container-Instance durchführen. Weitere Informationen finden Sie im Benutzerhandbuch für Application Load Balancers.

Informationen zu den bewährten Methoden für die Festlegung von Parametern zur Beschleunigung Ihrer Bereitstellungen finden Sie unter:

- Optimieren Sie die Parameter für die Zustandsprüfung des Load Balancers für Amazon ECS
- Optimieren Sie die Load Balancer-Verbindungsparameter für Amazon ECS

Beachten Sie Folgendes, wenn Sie Application Load Balancers mit Amazon ECS verwenden:

- Amazon ECS erfordert die service-verknüpfte IAM-Rolle, die die Berechtigungen bietet, die erforderlich sind, um Ziele bei Ihrem Load Balancer zu registrieren und abzumelden, wenn Aufgaben erstellt und gestoppt werden. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.
- Für die Zielgruppe muss der IP-Adresstyp auf IPv4 eingestellt sein.
- Für Services mit Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, müssen Sie beim Erstellen einer Zielgruppe für Ihren Service `ip` als Zieltyp auswählen, nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance.
- Wenn Ihr Dienst Zugriff auf mehrere Ports mit Lastenausgleich benötigt, z. B. Port 80 und Port 443 für einen HTTP/HTTPS-Dienst, können Sie zwei Listener konfigurieren. Ein Listener ist für HTTPS verantwortlich, sodass die Anforderung an den Service weitergeleitet wird, und ein anderer Listener für die Umleitung von HTTP-Anforderungen an den entsprechenden HTTPS-Port. Weitere Informationen finden Sie Erstellen eines Listeners für Ihren Application Load Balancer im Benutzerhandbuch für Application Load Balancers.
- Die Subnetzkonfiguration Ihres Load Balancers muss alle Availability Zones enthalten, in denen sich Ihre Container-Instances befinden.
- Nachdem Sie einen Dienst erstellt haben, kann die Load Balancer-Konfiguration von AWS Management Console nicht mehr geändert werden. Sie können den AWS Copilot AWS CLI oder das SDK verwenden AWS CloudFormation, um die Load Balancer-Konfiguration nur für den `ECS` Rolling Deployment Controller zu ändern, nicht für blau/grün oder extern. AWS CodeDeploy Wenn

Sie eine Konfiguration für den Load Balancer hinzufügen, aktualisieren oder entfernen, startet Amazon ECS eine neue Bereitstellung mit der aktualisierten Konfiguration Elastic Load Balancing. Dies führt dazu, dass sich Aufgaben bei Load Balancern registrieren und von diesen abmelden. Wir empfehlen, dass Sie dies in einer Testumgebung überprüfen, bevor Sie die Konfiguration Elastic Load Balancing Balancing aktualisieren. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz.

- Wenn eine Serviceaufgabe die Kriterien für die Zustandsprüfung des Load Balancers nicht erfüllt, wird die Aufgabe gestoppt und neu gestartet. Dieser Vorgang wird fortgesetzt, bis Ihr Service die Anzahl der gewünschten ausgeführten Aufgaben erreicht.
- Informationen zu Problemen mit Services, für die ein Load Balancer aktiviert ist, finden Sie unter Fehlerbehebung bei Service Load Balancers in Amazon ECS.
- Ihre Aufgaben und Ihr Load Balancer müssen sich in derselben VPC befinden.
- Verwenden Sie für jeden Service eine eindeutige Zielgruppe.

  Die Verwendung derselben Zielgruppe für mehrere Dienste kann zu Problemen bei der Bereitstellung von Diensten führen.

Informationen zum Erstellen eines Application Load Balancer finden Sie unter Erstellen eines Application Load Balancer in Application Load Balancers

Verwenden Sie einen Network Load Balancer für Amazon ECS

Ein Network Load Balancer trifft Routing-Entscheidungen auf der Transportebene (TCP/SSL). Er kann Millionen Anfragen pro Sekunde verarbeiten. Nachdem der Load Balancer eine Verbindung erhalten hat, wählt er mithilfe eines Flow-Hash-Routing-Algorithmus ein Ziel aus der Zielgruppe für die Standardregel aus. Er versucht, eine TCP-Verbindung zu dem ausgewählten Ziel auf dem in der Listener-Konfiguration angegebenen Port zu öffnen. Es leitet die Anforderung ohne Ändern der Headers weiter. Network Load Balancer unterstützen dynamische Host-Port-Zuweisung. Wenn die Containerdefinition Ihrer Aufgabe beispielsweise Port 80 als NGINX-Container-Port and Port 0 als Host-Port angibt, wird der Host-Port dynamisch aus dem flüchtigen Port-Bereich der Container-Instance ausgewählt (z. B. 32768 bis 61000 beim aktuellen Amazon-ECS-optimierten AMI). Beim Start der Aufgabe wird der NGINX-Container bei dem Network Load Balancer als Instance-ID- und Port-Kombination registriert, und der Datenverkehr wird an die Instance-ID und den Port verteilt, die diesem Container entsprechen. Aufgrund dieser dynamischen Zuordnung können Sie mehrere Aufgaben über einen einzelnen Service auf derselben Container-Instance durchführen. Weitere Informationen finden Sie im Benutzerhandbuch für Network Load Balancers.

Informationen zu den bewährten Methoden für die Festlegung von Parametern zur Beschleunigung Ihrer Bereitstellungen finden Sie unter:

- Optimieren Sie die Parameter für die Zustandsprüfung des Load Balancers für Amazon ECS
- Optimieren Sie die Load Balancer-Verbindungsparameter für Amazon ECS

Beachten Sie Folgendes, wenn Sie Network Load Balancers mit Amazon ECS verwenden:

- Amazon ECS erfordert die service-verknüpfte IAM-Rolle, die die Berechtigungen bietet, die erforderlich sind, um Ziele bei Ihrem Load Balancer zu registrieren und abzumelden, wenn Aufgaben erstellt und gestoppt werden. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.
- Sie können einem Service nicht mehr als fünf Zielgruppen zuordnen.
- Für Services mit Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, müssen Sie beim Erstellen einer Zielgruppe für Ihren Service `ip` als Zieltyp auswählen, nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance.
- Die Subnetzkonfiguration Ihres Load Balancers muss alle Availability Zones enthalten, in denen sich Ihre Container-Instances befinden.
- Nachdem Sie einen Dienst erstellt haben, kann die Load Balancer-Konfiguration von AWS Management Console nicht mehr geändert werden. Sie können den AWS Copilot AWS CLI oder das SDK verwenden AWS CloudFormation, um die Load Balancer-Konfiguration nur für den ECS Rolling Deployment Controller zu ändern, nicht für AWS CodeDeploy blau/grün oder extern. Wenn Sie eine Konfiguration für den Load Balancer hinzufügen, aktualisieren oder entfernen, startet Amazon ECS eine neue Bereitstellung mit der aktualisierten Konfiguration Elastic Load Balancing. Dies führt dazu, dass sich Aufgaben bei Load Balancern registrieren und von diesen abmelden. Wir empfehlen, dass Sie dies in einer Testumgebung überprüfen, bevor Sie die Konfiguration Elastic Load Balancing Balancing aktualisieren. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz.
- Wenn eine Serviceaufgabe die Kriterien für die Zustandsprüfung des Load Balancers nicht erfüllt, wird die Aufgabe gestoppt und neu gestartet. Dieser Vorgang wird fortgesetzt, bis Ihr Service die Anzahl der gewünschten ausgeführten Aufgaben erreicht.
- Wenn Sie einen Gateway Load Balancer verwenden, der mit IP-Adressen als Zielen konfiguriert ist und Client IP Preservation deaktiviert ist, werden Anfragen als von der privaten IP-Adresse des Gateway Load Balancers kommend angesehen. Das bedeutet, dass Dienste, die hinter

einem Gateway Load Balancer stehen, praktisch weltweit zugänglich sind, sobald Sie eingehende Anfragen und Zustandsprüfungen in der Zielsicherheitsgruppe zulassen.

- Für Fargate-Aufgaben müssen Sie die Plattformversion `1.4.0` (Linux) oder `1.0.0` (Windows) verwenden.
- Informationen zu Problemen mit Services, für die ein Load Balancer aktiviert ist, finden Sie unter Fehlerbehebung bei Service Load Balancers in Amazon ECS.
- Ihre Aufgaben und Ihr Load Balancer müssen sich in derselben VPC befinden.
- Die Erhaltung der IP-Adresse des Network Load Balancer Balancer-Clients ist mit Fargate-Zielen kompatibel.
- Verwenden Sie für jeden Dienst eine eigene Zielgruppe.

  Die Verwendung derselben Zielgruppe für mehrere Dienste kann zu Problemen bei der Bereitstellung von Diensten führen.

Informationen zum Erstellen eines Network Load Balancer finden Sie unter Erstellen eines Network Load Balancer in Network Load Balancers

> **Important**
>
> Wenn Ihre Service-Aufgabendefinition den Netzwerkmodus `awsvpc` verwendet (der für den Starttyp Fargate erforderlich ist), müssen Sie `ip` als Zieltyp verwenden, und nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance.
> Sie können Instances nicht nach Instance-ID registrieren, wenn sie die folgenden Instance-Typen haben: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 und T1. Sie können Instances dieser Arten nach IP-Adresse registrieren.

Verwenden Sie einen Gateway Load Balancer für Amazon ECS

Ein Gateway Load Balancer arbeitet auf der dritten Ebene des Open Systems Interconnection (OSI) -Modells, der Netzwerkschicht. Es überwacht alle IP-Pakete über alle Ports und leitet den Datenverkehr an die Zielgruppe weiter, die in der Listener-Regel angegeben ist. Es behält die Klebrigkeit der Flows zu einer bestimmten Ziel-Appliance mit 5-Tupel (für TCP/UDP-Flows) oder 3-Tupel (für Nicht-TCP/UDP-Flows) bei. Wenn die Containerdefinition Ihrer Aufgabe beispielsweise Port 80 als NGINX-Container-Port and Port 0 als Host-Port angibt, wird der Host-Port dynamisch

aus dem flüchtigen Port-Bereich der Container-Instance ausgewählt (z. B. 32768 bis 61000 beim aktuellen Amazon-ECS-optimierten AMI). Wenn die Aufgabe gestartet wird, wird der NGINX-Container beim Gateway Load Balancer als Kombination aus Instanz-ID und Port registriert, und der Verkehr wird an die Instanz-ID und den Port verteilt, die diesem Container entsprechen. Aufgrund dieser dynamischen Zuordnung können Sie mehrere Aufgaben über einen einzelnen Service auf derselben Container-Instance durchführen. Weitere Informationen finden Sie unter Was ist ein Gateway Load Balancer in Gateway Load Balancers.

Informationen zu den bewährten Methoden für die Festlegung von Parametern zur Beschleunigung Ihrer Bereitstellungen finden Sie unter:

- Optimieren Sie die Parameter für die Zustandsprüfung des Load Balancers für Amazon ECS
- Optimieren Sie die Load Balancer-Verbindungsparameter für Amazon ECS

Beachten Sie Folgendes, wenn Sie Gateway Load Balancers mit Amazon ECS verwenden:

- Amazon ECS erfordert die service-verknüpfte IAM-Rolle, die die Berechtigungen bietet, die erforderlich sind, um Ziele bei Ihrem Load Balancer zu registrieren und abzumelden, wenn Aufgaben erstellt und gestoppt werden. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.
- Für Services mit Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, müssen Sie beim Erstellen einer Zielgruppe für Ihren Service `ip` als Zieltyp auswählen, nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance.
- Die Subnetzkonfiguration Ihres Load Balancers muss alle Availability Zones enthalten, in denen sich Ihre Container-Instances befinden.
- Nachdem Sie einen Dienst erstellt haben, kann die Load Balancer-Konfiguration von AWS Management Console nicht mehr geändert werden. Sie können den AWS Copilot AWS CLI oder das SDK verwenden AWS CloudFormation, um die Load Balancer-Konfiguration nur für den `ECS` Rolling Deployment Controller zu ändern, nicht AWS CodeDeploy für blau/grün oder extern. Wenn Sie eine Konfiguration für den Load Balancer hinzufügen, aktualisieren oder entfernen, startet Amazon ECS eine neue Bereitstellung mit der aktualisierten Konfiguration Elastic Load Balancing. Dies führt dazu, dass sich Aufgaben bei Load Balancern registrieren und von diesen abmelden. Wir empfehlen, dass Sie dies in einer Testumgebung überprüfen, bevor Sie die Konfiguration Elastic Load Balancing Balancing aktualisieren. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz.

- Wenn eine Serviceaufgabe die Kriterien für die Zustandsprüfung des Load Balancers nicht erfüllt, wird die Aufgabe gestoppt und neu gestartet. Dieser Vorgang wird fortgesetzt, bis Ihr Service die Anzahl der gewünschten ausgeführten Aufgaben erreicht.
- Wenn Sie einen Gateway Load Balancer verwenden, der mit IP-Adressen als Ziele konfiguriert ist, werden Anfragen als von der privaten IP-Adresse des Gateway Load Balancers kommend angesehen. Das bedeutet, dass Dienste, die hinter einem Gateway Load Balancer stehen, praktisch weltweit zugänglich sind, sobald Sie eingehende Anfragen und Zustandsprüfungen in der Zielsicherheitsgruppe zulassen.
- Für Fargate-Aufgaben müssen Sie die Plattformversion `1.4.0` (Linux) oder `1.0.0` (Windows) verwenden.
- Informationen zu Problemen mit Services, für die ein Load Balancer aktiviert ist, finden Sie unter Fehlerbehebung bei Service Load Balancers in Amazon ECS.
- Ihre Aufgaben und Ihr Load Balancer müssen sich in derselben VPC befinden.
- Verwenden Sie für jeden Service eine eindeutige Zielgruppe.

  Die Verwendung derselben Zielgruppe für mehrere Dienste kann zu Problemen bei der Bereitstellung von Diensten führen.

Informationen zum Erstellen eines Gateway Load Balancer finden Sie unter Erstellen eines Gateway Load Balancer in Gateway Load Balancers

> **Important**
>
> Wenn Ihre Service-Aufgabendefinition den Netzwerkmodus `awsvpc` verwendet (der für den Starttyp Fargate erforderlich ist), müssen Sie `ip` als Zieltyp verwenden, und nicht `instance`. Das liegt daran, dass Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, mit einer Elastic-Network-Schnittstelle verknüpft sind, und nicht mit einer Amazon-EC2-Instance.
> Sie können Instances nicht nach Instance-ID registrieren, wenn sie die folgenden Instance-Typen haben: C1, CC1, CC2, CG1, CG2, CR1, G1, G2, HI1, HS1, M1, M2, M3 und T1. Sie können Instances dieser Arten nach IP-Adresse registrieren.

Registrierung mehrerer Zielgruppen bei einem Amazon ECS-Service

Ihr Amazon-ECS-Service kann den Datenverkehr von mehreren Load Balancern abwickeln und mehrere Ports mit Lastenausgleich festlegen, wenn Sie mehrere Zielgruppen in einer Servicedefinition angeben.

Um einen Service mit mehreren Zielgruppen zu erstellen, müssen Sie den Service mithilfe der Amazon ECS-API, des SDK oder einer AWS CloudFormation Vorlage erstellen. AWS CLI Nachdem der Service erstellt wurde, können Sie den Service und die Zielgruppen anzeigen, die in der AWS Management Console registriert sind. Sie müssen `UpdateService` verwenden, um die Load Balancer-Konfiguration eines vorhandenen Service zu ändern.

Mehrere Zielgruppen können in einer Servicedefinition angegeben werden, die das folgende Format verwendet. Die vollständige Syntax einer Servicedefinition finden Sie unter Servicedefinitionsvorlage.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"container_name",
      "containerPort":container_port
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"container_name",
      "containerPort":container_port
   }
]
```

Überlegungen

Folgendes sollte berücksichtigt werden, wenn Sie mehrere Zielgruppen in einer Servicedefinition angeben:

- Für Services, die einen Application Load Balancer oder einen Network Load Balancer verwenden, können Sie nicht mehr als fünf Zielgruppen an einen Service anfügen.
- Das Festlegen mehrerer Zielgruppen in einer Servicedefinition wird nur unter folgenden Bedingungen unterstützt:

  - Der Service muss entweder einen Application Load Balancer oder Network Load Balancer verwenden.
  - Der Service muss den Bereitstellungscontroller-Typ der laufenden Aktualisierung (ECS) haben.
- Die Angabe mehrerer Zielgruppen wird für Services mit Aufgaben unterstützt, die die Starttypen Fargate und EC2 verwenden.
- Wenn Sie einen Service erstellen, der mehrere Zielgruppen angibt, muss die serviceverknüpfte Amazon-ECS-Rolle erstellt werden. Die Rolle wird erstellt, indem der Parameter `role` in API-Anfragen oder die Eigenschaft `Role` in AWS CloudFormation weggelassen wird. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.

Beispiel für Service-Definitionen

Nachfolgend finden Sie einige Beispiel-Anwendungsfälle für das Festlegen mehrerer Zielgruppen in einer Servicedefinition. Die vollständige Syntax einer Servicedefinition finden Sie unter Servicedefinitionsvorlage.

Mit separaten Load Balancern für internen und externen Datenverkehr

Im folgenden Anwendungsfall verwendet ein Service zwei separate Load Balancer, einen für internen Datenverkehr und einen zweiten für Datenverkehr mit dem Internet für denselben Container und Port.

```
"loadBalancers":[
   //Internal ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"nginx",
      "containerPort":8080
   },
   //Internet-facing ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"nginx",
      "containerPort":8080
   }
]
```

Bereitstellung mehrerer Ports aus demselben Container

Im folgenden Anwendungsfall verwendet ein Service einen Load Balancer, stellt jedoch mehrere Ports vom selben Container bereit. Beispiel: Ein Jenkins-Container stellt Port 8080 für die Jenkins Web-Schnittstelle und Port 50000 für die API bereit.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"jenkins",
      "containerPort":8080
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"jenkins",
      "containerPort":50000
   }
]
```

Ports aus mehreren Containern verfügbar machen

Im folgenden Anwendungsfall verwendet ein Service einen Load Balancer und zwei Zielgruppen zur Bereitstellung aus separaten Container-Ports.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"webserver",
      "containerPort":80
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"database",
      "containerPort":3306
```

```
    }
]
```

Skalieren Sie Ihren Amazon ECS-Service automatisch

Auto Scaling ist die Fähigkeit, die gewünschte Anzahl Aufgaben in Ihrem Amazon-ECS-Service automatisch abhängig von der Nachfrage zu erhöhen oder zu verringern. Amazon ECS nutzt den Application Auto Scaling-Service, um diese Funktionalität bereitzustellen. Weitere Informationen finden Sie im Benutzerhandbuch zum Application Auto Scaling.

Amazon ECS veröffentlicht CloudWatch Metriken mit der durchschnittlichen CPU- und Speicherauslastung Ihres Services. Weitere Informationen finden Sie unter Kennzahlen zur Nutzung des Amazon ECS-Service. Sie können diese und andere CloudWatch Kennzahlen verwenden, um Ihren Service zu skalieren (mehr Aufgaben hinzufügen), um der hohen Nachfrage in Spitzenzeiten gerecht zu werden, und Ihren Service zu skalieren (weniger Aufgaben ausführen), um die Kosten in Zeiten geringer Auslastung zu senken.

Amazon-ECS-Service-Auto-Scaling unterstützt die folgenden Typen des Auto Scalings:

- Skalieren Sie Ihren Amazon ECS-Service mithilfe eines Zielmetrikwerts – Erhöhen oder verringern Sie die Anzahl der Aufgaben, die von Ihrem Service ausgeführt werden, auf Grundlage eines Zielwerts für eine bestimmte Metrik. Dies ähnelt der Art und Weise, wie ein Thermostat die Temperatur in Ihrem Zuhause konstant hält. Sie wählen eine Temperatur aus und der Thermostat erledigt den Rest.
- Skalieren Sie Ihren Amazon ECS-Service mithilfe vordefinierter Inkremente auf der Grundlage von Alarmen CloudWatch – Erhöhen oder verringern Sie die Anzahl der Aufgaben, die von Ihrem Service ausgeführt werden, auf Grundlage einer Gruppe von Skalierungsanpassungen, die als Schrittanpassungen bezeichnet werden und je nach Ausmaß der Alarmüberschreitung variieren.
- Skalieren Sie Ihren Amazon ECS-Service mithilfe eines Zeitplans— Erhöhen oder verringern Sie die Anzahl der Aufgaben, die Ihr Service ausführt, je nach Datum und Uhrzeit.

Überlegungen

Beachten Sie bei Verwendung von Skalierungsrichtlinien die folgenden Überlegungen:

- Amazon ECS sendet Metriken in 1-Minuten-Intervallen an CloudWatch. Metriken sind erst verfügbar, wenn die Cluster und Services die Metriken an sie senden CloudWatch, und Sie können keine CloudWatch Alarme für Metriken erstellen, die nicht existieren.

- Die Skalierungsrichtlinien unterstützen eine Ruhephase. Das ist die Anzahl Sekunden, für die darauf gewartet wird, dass eine vorherige Skalierungsaktivität wirksam wird.
  - Bei Scale-out-Ereignissen wird eine kontinuierliche (jedoch nicht übermäßige) horizontale Skalierung nach oben angestrebt. Nachdem Service Auto Scaling unter Verwendung einer Skalierungsrichtlinie erfolgreich horizontal nach oben skaliert hat, beginnt es mit der Berechnung der Ruhezeit. Die Skalierungsrichtlinie erhöht die gewünschte Kapazität nicht erneut, es sei denn, es wird eine größere Aufskalierung ausgelöst oder die Ruhephase endet. Während die Scale-Out-Ruhephase wirksam ist, wird die durch die initiierende horizontale Skalierung nach oben (Scale-Out) hinzugefügte Kapazität als Teil der gewünschten Kapazität für die nächste horizontale Skalierung nach oben berechnet.
  - Bei Scale-In-Ereignissen wird eine vorsichtige horizontale Skalierung nach unten angestrebt, um die Verfügbarkeit Ihrer Anwendung aufrechtzuerhalten, sodass horizontale Skalierungen nach unten blockiert werden, bis die Ruhephase abgelaufen ist. Wenn jedoch während der Abskalierungs-Ruhephase ein anderer Alarm eine Aufskalierungs-Aktivität auslöst, skaliert Service Auto Scaling das Ziel auf. In diesem Fall wird die Scale-In-Ruhephase angehalten und nicht abgeschlossen.
- Der Serviceplaner respektiert die gewünschte Anzahl zu jeder Zeit, aber solange Sie aktive Skalierungsrichtlinien und Alarme für einen Service haben, könnte Service Auto Scaling eine gewünschte Anzahl, die von Ihnen manuell festgelegt wurde, ändern.
- Wenn die gewünschte Anzahl eines Dienstes unter seinem Mindestkapazitätswert liegt und ein Alarm eine Scale-Out-Aktivität einleitet, skaliert Service Auto Scaling die gewünschte Anzahl bis zum Mindestkapazitätswert und skaliert dann nach Bedarf weiter, basierend auf der Skalierungsrichtlinie, die dem Alarm zugeordnet ist. Eine Herunterskalierung wird jedoch die gewünschte Anzahl nicht anpassen, da der Wert bereits unter dem Wert für die minimale Kapazität liegt.
- Wenn die gewünschte Anzahl eines Dienstes über seinem maximalen Kapazitätswert liegt und ein Alarm eine Aktivitätsskalierung einleitet, skaliert Service Auto Scaling die gewünschte Anzahl auf den maximalen Kapazitätswert und skaliert dann nach Bedarf weiter, basierend auf der Skalierungsrichtlinie, die dem Alarm zugeordnet ist. Eine Aufwärtsskalierung passt jedoch die gewünschte Anzahl nicht an, da der Wert bereits über dem Wert für die maximale Kapazität liegt.
- Während der Skalierungsaktivitäten ist die tatsächliche Anzahl der laufenden Aufgaben in einem Service der Wert, den Service Auto Scaling als Ausgangspunkt verwendet, im Gegensatz zu der gewünschten Anzahl. Dies ist, was die Verarbeitungskapazität sein soll. Dies verhindert eine übermäßige (unkontrollierte) Skalierung, die z. B. nicht erfüllt werden könnte, wenn nicht genügend Ressourcen für die Container Instances vorhanden sind, um die zusätzlichen Aufgaben

zu platzieren. Wenn die Container-Instance-Kapazität später verfügbar ist, kann die ausstehende Skalierung möglicherweise gelingen, und weitere Skalierungen können nach der Ruhephase erfolgen.

- Wenn Sie möchten, dass die Anzahl der Aufgaben auf Null skaliert wird, wenn es keine Arbeit zu erledigen gibt, legen Sie eine Mindestkapazität von 0 fest. Wenn die tatsächliche Kapazität 0 ist und die Metrik darauf hinweist, dass Workload-Bedarf besteht, wartet Service Auto Scaling, bis ein Datenpunkt gesendet wird, bevor die Skalierung ausgeführt wird. In diesem Fall skaliert es um den minimal möglichen Betrag als Startpunkt und setzt dann die Skalierung basierend auf der tatsächlichen Anzahl der laufenden Tasks fort.
- Application Auto Scaling deaktiviert Abskalierungsprozesse, während Amazon-ECS-Bereitstellungen ausgeführt werden. Scale-Out-Prozesse werden während einer Bereitstellung jedoch weiterhin ausgeführt, es sei denn, sie werden angehalten. Weitere Informationen finden Sie unter Auto Scaling und Bereitstellung von Services.
- Sie haben mehrere Application-Auto-Scaling-Optionen für Amazon-ECS-Aufgaben. Die Zielverfolgung ist der am einfachsten zu verwendende Modus. Damit müssen Sie lediglich einen Zielwert für eine Metrik festlegen, z. B. die durchschnittliche CPU-Auslastung. Anschließend verwaltet der Autoscaler automatisch die Anzahl der Aufgaben, die erforderlich sind, um diesen Wert zu erreichen. Mit der schrittweisen Skalierung können Sie schneller auf Bedarfsänderungen reagieren, da Sie die spezifischen Schwellenwerte für Ihre Skalierungsmetriken definieren und festlegen, wie viele Aufgaben hinzugefügt oder entfernt werden müssen, wenn die Schwellenwerte überschritten werden. Und was noch wichtiger ist: Sie können sehr schnell auf Änderungen der Nachfrage reagieren, indem Sie die Zeitspanne, in der ein Schwellenwertalarm überschritten wird, auf ein Minimum reduzieren.

Optimieren Sie die auto Skalierung des Amazon ECS-Service

Ein Amazon ECS-Service ist eine verwaltete Sammlung von Aufgaben. Jedem Service ist eine Aufgabendefinition, eine gewünschte Anzahl von Aufgaben und eine optionale Platzierungsstrategie zugeordnet. Amazon ECS Service Auto Scaling wird über den Application Auto Scaling-Service implementiert. Application Auto Scaling verwendet CloudWatch Metriken als Quelle für Skalierungsmetriken. Es verwendet auch CloudWatch Alarme, um Schwellenwerte dafür festzulegen, wann Ihr Service erweitert oder verkleinert werden soll. Sie geben die Schwellenwerte für die Skalierung an, indem Sie entweder ein metrisches Ziel festlegen, das als Zielverfolgungsskalierung bezeichnet wird, oder indem Sie Schwellenwerte angeben, die als schrittweise Skalierung bezeichnet werden. Nachdem Application Auto Scaling konfiguriert wurde, berechnet es kontinuierlich die

entsprechende gewünschte Task-Anzahl für den Service. Es benachrichtigt Amazon ECS auch, wenn sich die gewünschte Anzahl an Aufgaben ändern sollte, entweder durch Verkleinern oder Vergrößern.

Um Service Auto Scaling effektiv nutzen zu können, müssen Sie eine geeignete Skalierungsmetrik auswählen.

Eine Anwendung sollte horizontal skaliert werden, wenn prognostiziert wird, dass der Bedarf die aktuelle Kapazität übersteigt. Umgekehrt kann eine Anwendung skaliert werden, um Kosten zu sparen, wenn die Ressourcen den Bedarf übersteigen.

Identifizieren Sie eine Metrik

Für eine effektive Skalierung ist es wichtig, eine Kennzahl zu identifizieren, die auf Auslastung oder Sättigung hinweist. Diese Metrik muss die folgenden Eigenschaften aufweisen, um für die Skalierung nützlich zu sein.

- Die Metrik muss mit der Nachfrage korreliert sein. Wenn die Ressourcen konstant gehalten werden, sich die Nachfrage jedoch ändert, muss sich auch der Metrikwert ändern. Die Kennzahl sollte steigen oder sinken, wenn die Nachfrage steigt oder sinkt.
- Der metrische Wert muss proportional zur Kapazität skaliert werden. Wenn die Nachfrage konstant bleibt, muss das Hinzufügen weiterer Ressourcen zu einer proportionalen Änderung des metrischen Werts führen. Eine Verdoppelung der Anzahl der Aufgaben sollte also zu einer Verringerung der Metrik um 50% führen.

Der beste Weg, eine Nutzungsmetrik zu ermitteln, sind Belastungstests in einer Vorproduktionsumgebung, wie z. B. einer Staging-Umgebung. Kommerzielle und Open-Source-Lösungen für Lasttests sind weit verbreitet. Diese Lösungen können in der Regel entweder synthetische Last erzeugen oder echten Benutzerverkehr simulieren.

Um den Prozess der Belastungstests zu starten, erstellen Sie Dashboards für die Nutzungsmetriken Ihrer Anwendung. Zu diesen Kennzahlen gehören CPU-Auslastung, Speicherauslastung, I/O-Operationen, I/O-Warteschlangentiefe und Netzwerkdurchsatz. Sie können diese Metriken mit einem Service wie Container Insights sammeln. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights. Stellen Sie während dieses Vorgangs sicher, dass Sie Kennzahlen zu den Antwortzeiten oder den Abschlussquoten Ihrer Anwendung erfassen und grafisch darstellen.

Beginnen Sie mit einer kleinen Anfrage oder einer Stellenausschreibungsrate. Halten Sie diese Rate mehrere Minuten lang konstant, damit sich Ihre Bewerbung aufwärmen kann. Erhöhen Sie

dann langsam die Geschwindigkeit und halten Sie sie einige Minuten lang konstant. Wiederholen Sie diesen Zyklus und erhöhen Sie die Rate jedes Mal, bis die Antwort- oder Abschlusszeiten Ihrer Anwendung zu langsam sind, um Ihre Service Level Objectives (SLOs) zu erreichen.

Untersuchen Sie beim Auslastungstest die einzelnen Nutzungskennzahlen. Die Metriken, die mit der Auslastung steigen, eignen sich am besten als Ihre besten Nutzungskennzahlen.

Identifizieren Sie als Nächstes die Ressource, deren Sättigung erreicht ist. Untersuchen Sie gleichzeitig auch die Nutzungskennzahlen, um festzustellen, welche zuerst auf hohem Niveau abflacht oder einen Spitzenwert erreicht und dann Ihre Anwendung zuerst zum Absturz bringt. Wenn die CPU-Auslastung beispielsweise von 0 auf 70 bis 80% steigt, wenn Sie mehr Last hinzufügen, und dann auf diesem Niveau bleibt, wenn noch mehr Last hinzugefügt wird, dann kann man mit Sicherheit sagen, dass die CPU gesättigt ist. Je nach CPU-Architektur erreicht sie möglicherweise nie 100%. Gehen Sie beispielsweise davon aus, dass die Speicherauslastung zunimmt, wenn Sie mehr Last hinzufügen, und Ihre Anwendung dann plötzlich abstürzt, wenn sie das Speicherlimit für Aufgaben oder Amazon EC2 EC2-Instances erreicht. In dieser Situation ist es wahrscheinlich der Fall, dass der Speicher vollständig verbraucht wurde. Möglicherweise werden von Ihrer Anwendung mehrere Ressourcen verbraucht. Wählen Sie daher die Metrik aus, die die Ressource darstellt, die zuerst erschöpft ist.

Versuchen Sie abschließend erneut, die Auslastung zu testen, nachdem Sie die Anzahl der Aufgaben oder Amazon EC2 EC2-Instances verdoppelt haben. Gehen Sie davon aus, dass die Schlüsselkennzahl halb so schnell wie zuvor zunimmt oder abnimmt. Wenn dies der Fall ist, ist die Metrik proportional zur Kapazität. Dies ist eine gute Nutzungsmetrik für Auto Scaling.

Betrachten Sie nun dieses hypothetische Szenario. Nehmen wir an, Sie führen einen Lasttest für eine Anwendung durch und stellen fest, dass die CPU-Auslastung bei 100 Anfragen pro Sekunde irgendwann 80% erreicht. Wenn mehr Last hinzugefügt wird, erhöht sich die CPU-Auslastung nicht mehr. Dadurch reagiert Ihre Anwendung jedoch langsamer. Dann führen Sie den Auslastungstest erneut durch und verdoppeln dabei die Anzahl der Aufgaben, halten aber die Geschwindigkeit auf dem vorherigen Spitzenwert. Wenn Sie feststellen, dass die durchschnittliche CPU-Auslastung auf etwa 40% sinkt, ist die durchschnittliche CPU-Auslastung ein guter Kandidat für eine Skalierungsmetrik. Bleibt die CPU-Auslastung dagegen bei 80%, nachdem die Anzahl der Aufgaben erhöht wurde, ist die durchschnittliche CPU-Auslastung keine gute Skalierungsmetrik. In diesem Fall sind weitere Untersuchungen erforderlich, um eine geeignete Metrik zu finden.

Allgemeine Anwendungsmodelle und Skalierungseigenschaften

Es wird Software aller Art verwendet AWS. Viele Workloads sind selbst entwickelt, während andere auf beliebter Open-Source-Software basieren. Unabhängig davon, woher sie stammen, haben wir einige gängige Entwurfsmuster für Dienste beobachtet. Wie effektiv skaliert werden kann, hängt zu einem großen Teil vom Muster ab.

Der effiziente CPU-gebundene Server

Der effiziente CPU-gebundene Server nutzt fast keine anderen Ressourcen als den CPU- und Netzwerkdurchsatz. Jede Anfrage kann von der Anwendung alleine bearbeitet werden. Anfragen hängen nicht von anderen Diensten wie Datenbanken ab. Die Anwendung kann Hunderttausende von gleichzeitigen Anfragen verarbeiten und dafür mehrere CPUs effizient nutzen. Jede Anfrage wird entweder von einem dedizierten Thread mit geringem Speicheraufwand bedient, oder es gibt eine asynchrone Ereignisschleife, die auf jeder CPU läuft, die Anfragen bearbeitet. Jedes Replikat der Anwendung ist gleichermaßen in der Lage, eine Anfrage zu verarbeiten. Die einzige Ressource, die vor der CPU aufgebraucht sein könnte, ist die Netzwerkbandbreite. Bei CPU-gebundenen Diensten macht die Speicherauslastung selbst bei Spitzendurchsatz nur einen Bruchteil der verfügbaren Ressourcen aus.

Diese Art von Anwendung eignet sich für CPU-basiertes Auto-Scaling. Die Anwendung bietet maximale Flexibilität in Bezug auf die Skalierung. Es kann vertikal skaliert werden, indem größere Amazon EC2 EC2-Instances oder Fargate-vCPUs bereitgestellt werden. Und es kann auch horizontal skaliert werden, indem weitere Replikate hinzugefügt werden. Durch das Hinzufügen weiterer Replikate oder die Verdoppelung der Instanzgröße wird die durchschnittliche CPU-Auslastung im Verhältnis zur Kapazität um die Hälfte reduziert.

Wenn Sie Amazon EC2 EC2-Kapazität für diese Anwendung verwenden, sollten Sie erwägen, sie auf rechenoptimierten Instances wie der `c5` OR-Familie zu platzieren. `c6g`

Der effiziente speichergebundene Server

Der effiziente speichergebundene Server weist pro Anforderung eine beträchtliche Menge an Speicher zu. Bei maximaler Parallelität, aber nicht unbedingt bei Durchsatz, wird der Arbeitsspeicher erschöpft, bevor die CPU-Ressourcen aufgebraucht sind. Der einer Anforderung zugeordnete Speicher wird freigegeben, wenn die Anforderung endet. Zusätzliche Anfragen können akzeptiert werden, solange Speicherplatz verfügbar ist.

Diese Art von Anwendung eignet sich für speicherbasiertes Auto-Scaling. Die Anwendung bietet maximale Flexibilität in Bezug auf die Skalierung. Es kann sowohl vertikal skaliert werden, indem

ihm größere Amazon EC2- oder Fargate-Speicherressourcen zur Verfügung gestellt werden. Und es kann auch horizontal skaliert werden, indem weitere Replikate hinzugefügt werden. Durch das Hinzufügen weiterer Replikate oder die Verdoppelung der Instance-Größe kann die durchschnittliche Speicherauslastung im Verhältnis zur Kapazität um die Hälfte reduziert werden.

Wenn Sie Amazon EC2 EC2-Kapazität für diese Anwendung verwenden, sollten Sie erwägen, sie auf speicheroptimierten Instances wie der `r5` OR-Familie zu platzieren. `r6g`

Einige speichergebundene Anwendungen geben den Speicher, der einer Anforderung zugeordnet ist, nicht frei, wenn diese beendet ist, sodass eine Verringerung der Parallelität nicht zu einer Verringerung des verwendeten Speichers führt. Aus diesem Grund empfehlen wir nicht, die speicherbasierte Skalierung zu verwenden.

Der Worker-basierte Server

Der Worker-basierte Server verarbeitet nacheinander eine Anfrage für jeden einzelnen Worker-Thread. Bei den Worker-Threads kann es sich um Lightweight-Threads wie POSIX-Threads handeln. Sie können auch Threads mit höherem Gewicht sein, wie z. B. UNIX-Prozesse. Unabhängig davon, um welchen Thread es sich handelt, gibt es immer eine maximale Parallelität, die die Anwendung unterstützen kann. Normalerweise wird das Parallelitätslimit proportional zu den verfügbaren Speicherressourcen festgelegt. Wenn das Parallelitätslimit erreicht ist, werden zusätzliche Anfragen in eine Backlog-Warteschlange gestellt. Wenn die Backlog-Warteschlange überläuft, werden zusätzliche eingehende Anfragen sofort abgelehnt. Zu den gängigen Anwendungen, die diesem Muster entsprechen, gehören Apache Webserver und Gunicorn.

Die Parallelität von Anfragen ist normalerweise die beste Metrik für die Skalierung dieser Anwendung. Da es für jedes Replikat ein Limit für die Parallelität gibt, ist es wichtig, eine Skalierung vorzunehmen, bevor das durchschnittliche Limit erreicht wird.

Der beste Weg, Metriken zur Parallelität von Anfragen zu erhalten, besteht darin, sie von Ihrer Anwendung melden zu lassen. CloudWatch Jedes Replikat Ihrer Anwendung kann die Anzahl der gleichzeitigen Anfragen als benutzerdefinierte Metrik mit hoher Frequenz veröffentlichen. Wir empfehlen, die Frequenz auf mindestens einmal pro Minute einzustellen. Nachdem mehrere Berichte gesammelt wurden, können Sie die durchschnittliche Parallelität als Skalierungsmetrik verwenden. Diese Metrik wird berechnet, indem die gesamte Parallelität durch die Anzahl der Replikate dividiert wird. Wenn beispielsweise die Gesamtzahl der Parallelität 1000 beträgt und die Anzahl der Replikate 10 beträgt, beträgt die durchschnittliche Parallelität 100.

Wenn Ihre Anwendung hinter einem Application Load Balancer steht, können Sie die `ActiveConnectionCount` Metrik für den Load Balancer auch als Faktor in der Skalierungsmetrik

verwenden. Die `ActiveConnectionCount` Metrik muss durch die Anzahl der Replikate dividiert werden, um einen Durchschnittswert zu erhalten. Der Durchschnittswert muss für die Skalierung verwendet werden, nicht der Rohzählwert.

Damit dieses Design optimal funktioniert, sollte die Standardabweichung der Antwortlatenz bei niedrigen Anforderungsraten gering sein. Wir empfehlen, dass in Zeiten geringer Nachfrage die meisten Anfragen innerhalb kurzer Zeit beantwortet werden, und es gibt nicht viele Anfragen, deren Beantwortung deutlich länger als der Durchschnitt dauert. Die durchschnittliche Antwortzeit sollte nahe dem 95. Perzentil der Antwortzeit liegen. Andernfalls kann es zu Warteschlangenüberläufen kommen. Dies führt zu Fehlern. Wir empfehlen, dass Sie bei Bedarf zusätzliche Replikate bereitstellen, um das Risiko eines Überlaufs zu minimieren.

Der wartende Server

Der wartende Server verarbeitet jede Anfrage in gewissem Maße, ist jedoch in hohem Maße von einem oder mehreren nachgelagerten Diensten abhängig, damit er funktioniert. Containeranwendungen nutzen häufig stark nachgelagerte Dienste wie Datenbanken und andere API-Dienste. Es kann einige Zeit dauern, bis diese Dienste reagieren, insbesondere in Szenarien mit hoher Kapazität oder hoher Parallelität. Dies liegt daran, dass diese Anwendungen in der Regel nur wenige CPU-Ressourcen verwenden und gleichzeitig den verfügbaren Arbeitsspeicher maximal ausnutzen.

Der Wartedienst eignet sich entweder für das speichergebundene Servermuster oder das Worker-basierte Servermuster, je nachdem, wie die Anwendung konzipiert ist. Wenn die Parallelität der Anwendung nur durch den Arbeitsspeicher begrenzt ist, sollte die durchschnittliche Speicherauslastung als Skalierungsmetrik verwendet werden. Wenn die Parallelität der Anwendung auf einem Worker-Limit basiert, sollte die durchschnittliche Parallelität als Skalierungsmetrik verwendet werden.

Der Java-basierte Server

Wenn Ihr Java-basierter Server CPU-gebunden ist und proportional zu den CPU-Ressourcen skaliert, ist er möglicherweise für das effiziente CPU-gebundene Servermuster geeignet. Wenn das der Fall ist, könnte die durchschnittliche CPU-Auslastung als Skalierungsmetrik geeignet sein. Viele Java-Anwendungen sind jedoch nicht CPU-gebunden, was ihre Skalierung schwierig macht.

Für eine optimale Leistung empfehlen wir, dem Java Virtual Machine (JVM) -Heap so viel Speicher wie möglich zuzuweisen. Neuere Versionen der JVM, einschließlich Java 8 Update 191 oder höher, legen die Heap-Größe automatisch so groß wie möglich fest, damit sie in den Container passt. Das

bedeutet, dass in Java die Speicherauslastung selten proportional zur Anwendungsauslastung ist. Mit steigender Anforderungsrate und Parallelität bleibt die Speicherauslastung konstant. Aus diesem Grund empfehlen wir nicht, Java-basierte Server auf der Grundlage der Speicherauslastung zu skalieren. Stattdessen empfehlen wir in der Regel, je nach CPU-Auslastung zu skalieren.

In einigen Fällen kommt es bei Java-basierten Servern vor der CPU-Auslastung zu einer Heap-Erschöpfung. Wenn Ihre Anwendung bei hoher Parallelität zur Heap-Erschöpfung neigt, sind durchschnittliche Verbindungen die beste Skalierungsmetrik. Wenn Ihre Anwendung bei hohem Durchsatz zur Heap-Erschöpfung neigt, ist die durchschnittliche Anforderungsrate die beste Skalierungsmetrik.

Server, die andere Laufzeiten verwenden, bei denen Müll gesammelt wurde

Viele Serveranwendungen wie .NET und Ruby basieren auf Laufzeiten, die Garbage-Collection durchführen. Diese Serveranwendungen passen möglicherweise in eines der zuvor beschriebenen Muster. Wie bei Java empfehlen wir jedoch nicht, diese Anwendungen auf der Grundlage des Speichers zu skalieren, da ihre beobachtete durchschnittliche Speicherauslastung oft nicht mit dem Durchsatz oder der Parallelität korreliert.

Für diese Anwendungen empfehlen wir, die CPU-Auslastung entsprechend zu skalieren, wenn die Anwendung CPU-gebunden ist. Andernfalls empfehlen wir, die Skalierung auf der Grundlage Ihrer Lasttestergebnisse auf der Grundlage des durchschnittlichen Durchsatzes oder der durchschnittlichen Parallelität durchzuführen.

Jobprozessoren

Viele Workloads beinhalten asynchrone Auftragsverarbeitung. Dazu gehören Anwendungen, die Anfragen nicht in Echtzeit empfangen, sondern stattdessen eine Arbeitswarteschlange abonnieren, um Jobs zu erhalten. Für diese Art von Anwendungen ist die richtige Skalierungsmetrik fast immer die Warteschlangentiefe. Ein Anstieg der Warteschlange ist ein Hinweis darauf, dass die ausstehende Arbeit die Verarbeitungskapazität übersteigt, wohingegen eine leere Warteschlange darauf hindeutet, dass mehr Kapazität als zu erledigende Arbeit vorhanden ist.

AWS Messaging-Dienste wie Amazon SQS und Amazon Kinesis Data Streams bieten CloudWatch Metriken, die für die Skalierung verwendet werden können. Für Amazon SQS `ApproximateNumberOfMessagesVisible` ist dies die beste Metrik. Für Kinesis Data Streams sollten Sie die `MillisBehindLatest` Metrik verwenden, die von der Kinesis Client Library (KCL) veröffentlicht wurde. Diese Metrik sollte für alle Verbraucher gemittelt werden, bevor sie für die Skalierung verwendet wird.

Auto Scaling und Bereitstellung von Services

Application Auto Scaling deaktiviert Abskalierungsprozesse, während Amazon-ECS-Bereitstellungen ausgeführt werden. Scale-Out-Prozesse werden während einer Bereitstellung jedoch weiterhin ausgeführt, es sei denn, sie werden angehalten. Wenn Sie Scale-Out-Prozesse anhalten möchten, während Bereitstellungen ausgeführt werden, führen Sie die folgenden Schritte aus.

1. Rufen Sie den Befehl describe-scalable-targets (Beschreibe skalierbare Ziele) auf, indem Sie die Ressourcen-ID des ECS-Services angeben, der dem skalierbaren Ziel in Application Auto Scaling zugeordnet ist (Beispiel: `service/default/sample-webapp`). Zeichnen Sie die Ausgabe auf. Sie werden diese benötigen, wenn Sie den nächsten Befehl aufrufen.
2. Rufen Sie den Befehl register-scalable-target auf, indem Sie die Ressourcen-ID, den Namespace und die skalierbare Dimension angeben. Geben Sie `true` für sowohl `DynamicScalingInSuspended` als auch `DynamicScalingOutSuspended` an.
3. Nach Abschluss der Bereitstellung können Sie den Befehl register-scalable-target aufrufen, um die Skalierung fortzusetzen.

Weitere Informationen finden Sie unter Unterbrechen und Wiederaufnehmen der Skalierung für Application Auto Scaling.

Skalieren Sie Ihren Amazon ECS-Service mithilfe eines Zielmetrikwerts

Mit Skalierungsrichtlinien für die Zielverfolgung wählen Sie eine Metrik aus und legen einen Zielwert fest. Amazon ECS Service Auto Scaling erstellt und verwaltet die CloudWatch Alarme, die die Skalierungsrichtlinie steuern, und berechnet die Skalierungsanpassung auf der Grundlage der Metrik und des Zielwerts. Durch die Skalierungsrichtlinie werden so viele Service-Aufgaben wie erforderlich hinzugefügt oder entfernt, damit die Metrik auf oder nahe an dem Zielwert gehalten wird. Abgesehen davon, dass eine Skalierungsrichtlinie für die Ziel-Nachverfolgung die Metrik nahe an dem Zielwert hält, passt sie sich auch an die Schwankungen in der Metrik aufgrund eines schwankenden Lastmusters an und verringert schnelle Schwankungen der Anzahl der Aufgaben, die in Ihrem Service ausgeführt werden.

Überlegungen

Berücksichtigen Sie bei der Verwendung von Zielverfolgungsrichtlinien Folgendes:

- Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung geht davon aus, dass sie eine horizontale Skalierung nach oben vornehmen soll, wenn die angegebene Metrik über dem Zielwert liegt. Sie

können keine Skalierungsrichtlinie für die Ziel-Nachverfolgung verwenden, um eine horizontale Skalierung nach oben vorzunehmen, wenn die angegebene Metrik unter dem Zielwert liegt.

- Eine Skalierungsrichtlinie für die Ziel-Nachverfolgung nimmt keine Skalierung vor, wenn die angegebene Metrik unzureichende Daten aufweist. Es wird keine horizontale Skalierung nach unten vorgenommen, da unzureichende Daten nicht als geringe Auslastung interpretiert werden.
- Möglicherweise werden Lücken zwischen den Datenpunkten für den Zielwert und die aktuelle Metrik angezeigt. Der Grund hierfür ist, dass Service Auto Scaling stets vorsichtig agiert, indem beim Ermitteln der hinzuzufügenden oder zu entfernenden Kapazität Auf- oder Abrundungen vorgenommen werden. Dadurch wird verhindert, dass zu wenig Kapazität hinzufügt oder zu viel Kapazität entfernt wird.
- Um die Verfügbarkeit der Anwendung sicherzustellen, wird der Service schnellstmöglich proportional zur Metrik hochskaliert, jedoch etwas langsamer herunterskaliert.
- Application Auto Scaling deaktiviert Abskalierungsprozesse, während Amazon-ECS-Bereitstellungen ausgeführt werden. Scale-Out-Prozesse werden während einer Bereitstellung jedoch weiterhin ausgeführt, es sei denn, sie werden angehalten. Weitere Informationen finden Sie unter Auto Scaling und Bereitstellung von Services.
- Sie können mehrere Skalierungsrichtlinien für die Ziel-Nachverfolgung für einen Amazon-ECS-Service erstellen, vorausgesetzt, dass diese alle verschiedene Metriken verwenden. Da Service Auto Scaling immer darauf ausgerichtet ist, Verfügbarkeit zu priorisieren, ist sein Verhalten davon abhängig, ob die Richtlinien für die Ziel-Nachverfolgung für die horizontale Skalierung nach oben oder nach unten bereit sind. Sofern Richtlinien für die Ziel-Nachverfolgung für die Aufskalierung bereit sind, findet eine Aufskalierung des Services statt. Eine Abskalierung wird jedoch nur durchgeführt, wenn alle Richtlinien für die Ziel-Nachverfolgung (mit aktivierter Abskalierung) zur Abskalierung bereit sind.
- Bearbeiten oder löschen Sie nicht die CloudWatch Alarme, die Service Auto Scaling für eine Skalierungsrichtlinie zur Zielverfolgung verwaltet. Service Auto Scaling löscht die Alarme automatisch, wenn Sie die Skalierungsrichtlinie löschen.
- Die `ALBRequestCountPerTarget`-Metrik für die Skalierungsrichtlinien für Zielverfolgung wird für den Blau/Grün-Bereitstellungstyp nicht unterstützt.

Weitere Informationen zu Skalierungsrichtlinien für die Ziel-Nachverfolgung finden Sie unter Skalierungsrichtlinien für die Ziel-Nachverfolgung im Benutzerhandbuch zum Auto Scaling von Anwendungen.

So konfigurieren Sie Zielskalierungsrichtlinien für Ihren Amazon ECS-Service mithilfe der Amazon ECS-Konsole

1. Zusätzlich zu den standardmäßigen IAM-Berechtigungen für die Erstellung und Aktualisierung von Services benötigen Sie zusätzliche Berechtigungen. Weitere Informationen finden Sie unter Für Amazon ECS Service Auto Scaling sind IAM-Berechtigungen erforderlich.
2. Sie können eine Skalierungsrichtlinie konfigurieren, wenn Sie einen Dienst erstellen oder aktualisieren. Weitere Informationen finden Sie unter einem der folgenden Themen:
   - Erstellen eines Services mit definierten Parametern— Erstellen Sie einen neuen Dienst
   - Aktualisieren eines Amazon ECS-Service mithilfe der Konsole— Aktualisiere einen bestehenden Dienst

Um Richtlinien für die Zielskalierung für Ihren Amazon ECS-Service zu konfigurieren, verwenden Sie den AWS CLI

1. Zusätzlich zu den standardmäßigen IAM-Berechtigungen für die Erstellung und Aktualisierung von Services benötigen Sie zusätzliche Berechtigungen. Weitere Informationen finden Sie unter Für Amazon ECS Service Auto Scaling sind IAM-Berechtigungen erforderlich.
2. Registrieren Sie die Ihren Amazon-ECS-Service mit dem Befehl register-scalable-target als skalierbares Ziel.
3. Erstellen Sie eine Skalierungsrichtlinie mit dem Befehl put-scaling-policy.

Skalieren Sie Ihren Amazon ECS-Service mithilfe vordefinierter Inkremente auf der Grundlage von Alarmen CloudWatch

Mit Richtlinien zur schrittweisen Skalierung geben Sie CloudWatch Alarme an, die den Skalierungsprozess einleiten. Wenn Sie beispielsweise skalieren möchten, wenn die CPU-Auslastung ein bestimmtes Niveau erreicht, erstellen Sie anhand der bereitgestellten `CPUUtilization` Metrik einen Alarm. Beim Erstellen einer Richtlinie zur schrittweisen Skalierung müssen Sie einen der folgenden Skalierungsanpassungstypen angeben:

- Hinzufügen — Erhöhen Sie die Anzahl der Aufgaben um eine bestimmte Anzahl von Kapazitätseinheiten oder um einen bestimmten Prozentsatz der aktuellen Kapazität.
- Entfernen — Verringern Sie die Anzahl der Aufgaben um eine bestimmte Anzahl von Kapazitätseinheiten oder um einen bestimmten Prozentsatz der aktuellen Kapazität.

- Einstellen auf — Legt die Anzahl der Aufgaben auf die angegebene Anzahl von Kapazitätseinheiten fest.

Nehmen wir beispielsweise an, dass die Zielkapazität und die erfüllte Kapazität 10 betragen und die Skalierungsrichtlinie 1 hinzufügt. Wenn der Alarm verletzt wird, addiert der automatische Skalierungsprozess 1 zu 10 und ergibt 11, sodass Amazon ECS eine Aufgabe für den Service startet.

Wir empfehlen dringend, Skalierungsrichtlinien für die Zielverfolgung zu verwenden, um anhand von Kennzahlen wie der durchschnittlichen CPU-Auslastung oder der durchschnittlichen Anzahl von Anfragen pro Ziel zu skalieren. Metriken, die bei steigender Kapazität abnehmen und bei sinkender Kapazität zunehmen, können verwendet werden, um mithilfe von Target Tracking die Anzahl der Aufgaben proportional zu skalieren oder zu erhöhen. Dadurch wird sichergestellt, dass Service Auto Scaling der Nachfragekurve für Ihre Anwendungen genau folgt.

Einen Überblick über Step Scaling-Richtlinien und deren Funktionsweise finden Sie unter Step Scaling-Richtlinien im Application Auto Scaling Scaling-Benutzerhandbuch. Nachdem Sie diese Einführung gelesen haben, erfahren Sie in den folgenden Abschnitten, wie Sie Step Scaling für Amazon ECS mithilfe der Konsole und konfigurieren AWS Command Line Interface.

So konfigurieren Sie Step Scaling-Richtlinien für Ihren Amazon ECS-Service mithilfe der Amazon ECS-Konsole

1. Zusätzlich zu den standardmäßigen IAM-Berechtigungen für die Erstellung und Aktualisierung von Services benötigen Sie zusätzliche Berechtigungen. Weitere Informationen finden Sie unter Für Amazon ECS Service Auto Scaling sind IAM-Berechtigungen erforderlich.
2. Sie können eine Skalierungsrichtlinie konfigurieren, wenn Sie einen Dienst erstellen oder aktualisieren. Weitere Informationen finden Sie unter einem der folgenden Themen:
   - Erstellen eines Services mit definierten Parametern— Erstellen Sie einen neuen Dienst
   - Aktualisieren eines Amazon ECS-Service mithilfe der Konsole— Aktualisiere einen bestehenden Dienst

Um Step Scaling-Richtlinien für Ihren Amazon ECS-Service zu konfigurieren, verwenden Sie den AWS CLI

1. Zusätzlich zu den standardmäßigen IAM-Berechtigungen für die Erstellung und Aktualisierung von Services benötigen Sie zusätzliche Berechtigungen. Weitere Informationen finden Sie unter Für Amazon ECS Service Auto Scaling sind IAM-Berechtigungen erforderlich.

2. Registrieren Sie die Ihren Amazon-ECS-Service mit dem Befehl register-scalable-target als skalierbares Ziel.
3. Erstellen Sie eine Skalierungsrichtlinie mit dem Befehl put-scaling-policy.
4. Erstellen Sie mit dem Befehl put-metric-alarm einen Alarm, der die Skalierungsrichtlinie initiiert.

Skalieren Sie Ihren Amazon ECS-Service mithilfe eines Zeitplans

Mit der geplanten Skalierung können Sie eine automatische Skalierung für Ihre Anwendung auf der Grundlage vorhersehbarer Laständerungen einrichten, indem Sie geplante Aktionen erstellen, mit denen die Kapazität zu bestimmten Zeiten erhöht oder verringert wird. So können Sie Ihre Anwendung proaktiv skalieren, um sie an vorhersehbare Laständerungen anzupassen.

Mit diesen geplanten Skalierungsaktionen können Sie Kosten und Leistung optimieren. Ihre Anwendung verfügt über ausreichend Kapazität, um die Hauptverkehrsspitzen unter der Woche zu bewältigen, ohne dass zu anderen Zeiten unnötige Kapazitäten bereitgestellt werden.

Sie können geplante Skalierung und Skalierungsrichtlinien zusammen verwenden, um die Vorteile von proaktiven und reaktiven Skalierungsansätzen zu nutzen. Nachdem eine geplante Skalierungsaktion ausgeführt wurde, kann die Skalierungsrichtlinie weiterhin Entscheidungen darüber treffen, ob die Kapazität weiter skaliert werden soll. So können Sie sicherstellen, dass Sie über eine ausreichende Kapazität verfügen, um die Last für Ihre Anwendung zu bewältigen. Während sich Ihre Anwendung an die Nachfrage anpasst, muss die aktuelle Kapazität innerhalb der minimalen und maximalen Kapazität liegen, die durch Ihre geplante Aktion festgelegt wurde.

Sie können die Zeitplanskalierung mit dem konfigurieren AWS CLI. Weitere Informationen zur geplanten Skalierung finden Sie unter Scheduled Scaling im Application Auto Scaling Scaling-Benutzerhandbuch.

Amazon ECS-Services verbinden

Anwendungen, die in Amazon-ECS-Aufgaben ausgeführt werden, müssen häufig Verbindungen aus dem Internet empfangen oder eine Verbindung zu anderen Anwendungen, die in Amazon-ECS-Services ausgeführt werden, herstellen. Wenn Sie externe Verbindungen aus dem Internet benötigen, empfehlen wir die Verwendung von Elastic Load Balancing. Weitere Informationen zu integriertem Load Balancing finden Sie unter the section called “Verwenden Sie Load Balancing, um den Servicedatenverkehr zu verteilen”.

Wenn Sie eine Anwendung benötigen, um eine Verbindung zu anderen Anwendungen herzustellen, die in Amazon-ECS-Services ausgeführt werden, bietet Amazon ECS die folgenden Möglichkeiten, dies ohne einen Load Balancer zu tun:

- Amazon ECS Service Connect

  Wir empfehlen Service Connect, das eine Amazon ECS-Konfiguration für Serviceerkennung, Konnektivität und Verkehrsüberwachung bietet. Mit Service Connect können Ihre Anwendungen Kurznamen und Standardports verwenden, um eine Verbindung zu Amazon ECS-Services im selben Cluster oder in anderen Clustern herzustellen, auch über VPCs in demselben AWS-Region.

  Wenn Sie Service Connect verwenden, verwaltet Amazon ECS alle Teile der Serviceerkennung: die Erstellung der Namen, die erkannt werden können, die dynamische Verwaltung von Einträgen für jede Aufgabe, wenn die Aufgaben gestartet und beendet werden, das Ausführen eines Agenten in jeder Aufgabe, der für die Erkennung der Namen konfiguriert ist. Ihre Anwendung kann die Namen mithilfe der Standardfunktionen für DNS-Namen und das Herstellen von Verbindungen nachschlagen. Wenn Ihre Anwendung dies bereits tut, müssen Sie Ihre Anwendung nicht ändern, um Service Connect verwenden zu können.

  Sie stellen die vollständige Konfiguration in jeder Dienst- und Aufgabendefinition bereit. Amazon ECS verwaltet Änderungen an dieser Konfiguration in jeder Servicebereitstellung, um sicherzustellen, dass sich alle Aufgaben in einer Bereitstellung auf die gleiche Weise verhalten. Ein häufiges Problem bei DNS as Service Discovery ist beispielsweise die Steuerung einer Migration. Wenn Sie einen DNS-Namen so ändern, dass er auf die neuen Ersatz-IP-Adressen verweist, kann es die maximale TTL-Zeit dauern, bis alle Clients den neuen Dienst verwenden. Mit Service Connect aktualisiert die Client-Bereitstellung die Konfiguration, indem sie die Client-Tasks ersetzt. Sie können den Deployment Circuit Breaker und andere Bereitstellungskonfigurationen so konfigurieren, dass sie Service Connect-Änderungen genauso beeinflussen wie jede andere Bereitstellung.

  Weitere Informationen finden Sie unter Verwenden Sie Service Connect, um Amazon ECS-Services mit Kurznamen zu verbinden.

- Amazon-ECS-Serviceerkennung

  Ein weiterer service-to-service Kommunikationsansatz ist die direkte Kommunikation mithilfe von Service Discovery. Bei diesem Ansatz können Sie die AWS Cloud Map Service Discovery-Integration mit Amazon ECS verwenden. Mithilfe von Service Discovery synchronisiert Amazon ECS die Liste der gestarteten Aufgaben mit AWS Cloud Map, das einen DNS-Hostnamen

verwaltet, der in die internen IP-Adressen einer oder mehrerer Aufgaben von diesem bestimmten Service aufgelöst wird. Andere Dienste in der Amazon VPC können diesen DNS-Hostnamen verwenden, um Traffic mithilfe seiner internen IP-Adresse direkt an einen anderen Container zu senden.

Dieser service-to-service Kommunikationsansatz bietet eine geringe Latenz. Zwischen den Containern befinden sich keine zusätzlichen Komponenten. Der Verkehr wird direkt von einem Container zum anderen Container geleitet.

Dieser Ansatz eignet sich für den `awsvpc` Netzwerkmodus, in dem jede Aufgabe ihre eigene eindeutige IP-Adresse hat. Die meiste Software unterstützt nur die Verwendung von `A` DNS-Einträgen, die direkt in IP-Adressen aufgelöst werden. Bei Verwendung des `awsvpc` Netzwerkmodus handelt es sich bei den IP-Adressen für jede Aufgabe um einen `A` Datensatz. Wenn Sie jedoch den `bridge` Netzwerkmodus verwenden, können sich mehrere Container dieselbe IP-Adresse teilen. Darüber hinaus führen dynamische Portzuordnungen dazu, dass den Containern nach dem Zufallsprinzip Portnummern für diese einzelne IP-Adresse zugewiesen werden. Zu diesem Zeitpunkt reicht ein `A` Datensatz für die Diensterkennung nicht mehr aus. Sie müssen auch einen `SRV` Datensatz verwenden. Dieser Datensatztyp kann sowohl IP-Adressen als auch Portnummern nachverfolgen, erfordert jedoch, dass Sie die Anwendungen entsprechend konfigurieren. Einige vorgefertigte Anwendungen, die Sie verwenden, unterstützen möglicherweise keine `SRV` Datensätze.

Ein weiterer Vorteil des `awsvpc` Netzwerkmodus besteht darin, dass Sie für jeden Dienst eine eigene Sicherheitsgruppe haben. Sie können diese Sicherheitsgruppe so konfigurieren, dass eingehende Verbindungen nur von den spezifischen Upstream-Diensten zugelassen werden, die mit diesem Dienst kommunizieren müssen.

Der Hauptnachteil der direkten service-to-service Kommunikation mithilfe von Service Discovery besteht darin, dass Sie zusätzliche Logik implementieren müssen, um Wiederholungsversuche durchzuführen und Verbindungsfehler zu beheben. DNS-Einträge haben einen Zeitraum time-to-live (TTL), der bestimmt, wie lange sie zwischengespeichert werden. Es dauert einige Zeit, bis der DNS-Eintrag aktualisiert ist und der Cache abläuft, sodass Ihre Anwendungen die neueste Version des DNS-Eintrags abrufen können. Daher löst Ihre Anwendung den DNS-Eintrag möglicherweise so auf, dass er auf einen anderen Container verweist, der nicht mehr vorhanden ist. Ihre Anwendung muss Wiederholungsversuche verarbeiten und über eine Logik verfügen, um fehlerhafte Backends zu ignorieren.

Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.

Kompatibilitätstabelle für den Netzwerkmodus

In der folgenden Tabelle wird die Kompatibilität zwischen diesen Optionen und den Aufgaben-Netzwerkmodi beschrieben. In der Tabelle bezieht sich „Client“ auf die Anwendung, die die Verbindungen innerhalb einer Amazon-ECS-Aufgabe herstellt.

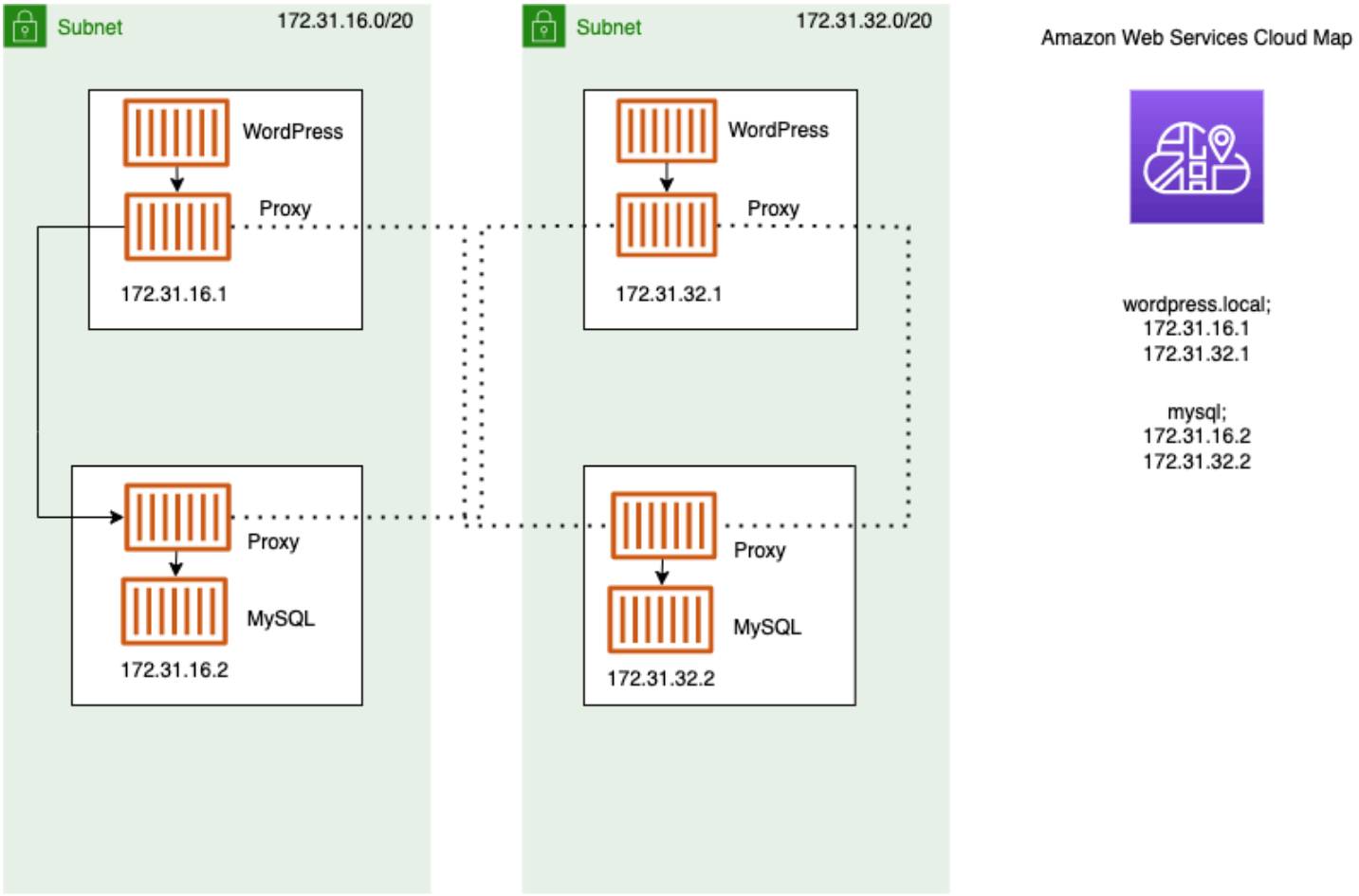
Verbindungsoptionen	Überbrückt	**`awsvpc`**	Host
Serviceerkennung	Ja, setzt jedoch voraus, dass Clients die SRV-Einträge im DNS ohne `hostPort` kennen.	Ja	Ja, setzt jedoch voraus, dass Clients die SRV-Einträge im DNS ohne `hostPort` kennen.
Service Connect	Ja	Ja	Nein

Verwenden Sie Service Connect, um Amazon ECS-Services mit Kurznamen zu verbinden

Amazon ECS Service Connect ermöglicht die Verwaltung der service-to-service Kommunikation als Amazon ECS-Konfiguration. Es erstellt sowohl Service Discovery als auch ein Service Mesh in Amazon ECS. Dies bietet die vollständige Konfiguration in jedem Dienst, den Sie über Dienstbereitstellungen verwalten, eine einheitliche Methode, um auf Ihre Dienste innerhalb von Namespaces zu verweisen, die nicht von der VPC-DNS-Konfiguration abhängt, sowie standardisierte Metriken und Protokolle zur Überwachung all Ihrer Anwendungen. Service Connect verbindet nur Dienste miteinander.

Das folgende Diagramm zeigt ein Beispiel für ein Service-Connect-Netzwerk mit 2 Subnetzen in der VPC und 2 Services. Ein Client-Dienst, der WordPress mit einer Aufgabe in jedem Subnetz ausgeführt wird. Ein Serverservice, der MySQL mit 1 Aufgabe in jedem Subnetz ausführt. Beide Services sind hochverfügbar und widerstandsfähig gegenüber Aufgaben- und Availability-Zone-Problemen, da jeder Service mehrere Aufgaben ausführt, die auf zwei Subnetze verteilt sind. Die

durchgezogenen Pfeile zeigen eine Verbindung von WordPress zu MySQL. Zum Beispiel ein `mysql --host=mysql` CLI-Befehl, der innerhalb des WordPress Containers in der Aufgabe mit der IP-Adresse ausgeführt wird`172.31.16.1`. Der Befehl verwendet den Kurznamen `mysql` auf dem Standardport für MySQL. Dieser Name und dieser Port stellen eine Verbindung zum Service-Connect-Proxy in derselben Aufgabe her. Der Proxy in der WordPress Aufgabe verwendet Round-Robin-Load Balancing und alle früheren Fehlerinformationen bei der Ausreißererkennung, um auszuwählen, zu welcher MySQL-Aufgabe eine Verbindung hergestellt werden soll. Wie die ausgefüllten Pfeile im Diagramm zeigen, stellt der Proxy eine Verbindung mit dem zweiten Proxy in der MySQL-Aufgabe mit der IP-Adresse `172.31.16.2` her. Der zweite Proxy stellt in derselben Aufgabe eine Verbindung zum lokalen MySQL-Server her. Beide Proxys melden die Verbindungsleistung, die in Diagrammen in den Amazon ECS- und CloudWatch Amazon-Konsolen sichtbar ist, sodass Sie Leistungskennzahlen von allen Arten von Anwendungen auf dieselbe Weise abrufen können.



Die folgenden Begriffe werden mit Service Connect verwendet.

Port-Name

Die Konfiguration der Amazon-ECS-Aufgabendefinition, die einer bestimmten Portzuordnung einen Namen zuweist. Diese Konfiguration wird nur von Amazon ECS Service Connect verwendet.

Client-Alias

Die Amazon-ECS-Service-Konfiguration, die die im Endpunkt verwendete Portnummer zuweist. Darüber hinaus kann der Client-Alias den DNS-Namen des Endpunkts zuweisen und den Erkennungsnamen überschreiben. Wenn im Amazon-ECS-Service kein Erkennungsname angegeben ist, überschreibt der Client-Aliasname den Portnamen als Endpunktnamen. Beispiele für Endpunkte finden Sie in der Definition von Endpunkt. Einem Amazon-ECS-Service können mehrere Client-Aliase zugewiesen werden. Diese Konfiguration wird nur von Amazon ECS Service Connect verwendet.

Erkennungsname

Der optionale Zwischenname, den Sie für einen bestimmten Port aus der Aufgabendefinition erstellen können. Dieser Name wird verwendet, um einen AWS Cloud Map Service zu erstellen. Wenn dieser Name nicht angegeben wird, wird der Portname aus der Aufgabendefinition verwendet. Einem bestimmten Port und Amazon-ECS-Service können mehrere Erkennungsnamen zugewiesen werden. Diese Konfiguration wird nur von Amazon ECS Service Connect verwendet.

AWS Cloud Map Dienstnamen müssen innerhalb eines Namespaces eindeutig sein. Aufgrund dieser Einschränkung können Sie für eine bestimmte Aufgabendefinition in jedem Namespace nur eine Service-Connect-Konfiguration ohne Erkennungsnamen haben.

Endpunkt

Die URL zum Herstellen einer Verbindung mit einer API oder Website. Die URL enthält das Protokoll, einen DNS-Namen und den Port. Für weitere Informationen über Endpunkte im Allgemeinen siehe Endpunkt im AWS -Glossary in der Allgemeine Amazon Web Services-Referenz.

Service Connect erstellt Endpunkte, die eine Verbindung zu Amazon-ECS-Services herstellen, und konfiguriert die Aufgaben in Amazon-ECS-Services, um eine Verbindung zu den Endpunkten herzustellen. Die URL enthält das Protokoll, einen DNS-Namen und den Port. Sie wählen das Protokoll und den Portnamen in der Aufgabendefinition aus, da der Port mit der Anwendung übereinstimmen muss, die sich im Container-Image befindet. Im Service wählen Sie jeden

Port nach Namen aus und können den DNS-Namen zuweisen. Wenn Sie in der Amazon-ECS-Servicekonfiguration keinen DNS-Namen angeben, wird standardmäßig der Portname aus der Aufgabendefinition verwendet. Ein Service-Connect-Endpunkt könnte beispielsweise `http://blog:80`, `grpc://checkout:8080` oder `http://_db.production.internal:99` sein.

Service-Connect-Service

Die Konfiguration eines einzelnen Endpunkts in einem Amazon-ECS-Service. Dies ist ein Teil der Service Connect-Konfiguration, bestehend aus einer einzelnen Zeile in der Service Connect and discovery name configuration (Service-Connect- und Erkennungsnamen-Konfiguration) in der Konsole oder einem Objekt in der `services`-Liste in der JSON-Konfiguration eines Amazon ECS-Service. Diese Konfiguration wird nur von Amazon ECS Service Connect verwendet.

Weitere Informationen finden Sie unter ServiceConnectService in der Amazon Elastic Container Service API-Referenz.

Namespace

Der Kurzname oder der vollständige Amazon Resource Name (ARN) des AWS Cloud Map Namespaces zur Verwendung mit Service Connect. Der Namespace muss mit dem Amazon ECS-Service und -Cluster identisch AWS-Region sein. Der Typ des AWS Cloud Map Namespaces in hat keinen Einfluss auf Service Connect.

Service Connect verwendet den AWS Cloud Map Namespace als logische Gruppierung von Amazon ECS-Aufgaben, die miteinander kommunizieren. Jeder Amazon-ECS-Service kann nur einem Namespace angehören. Die Dienste innerhalb eines Namespace können auf verschiedene Amazon ECS-Cluster innerhalb desselben AWS-Region verteilt werden. AWS-Konto Sie können Dienste nach beliebigen Kriterien frei organisieren.

Client-Service

Ein Dienst, der eine Netzwerk-Client-Anwendung ausführt. Für diesen Service muss ein Namespace konfiguriert sein. Jede Aufgabe im Service kann alle Endpunkte im Namespace über einen Service-Connect-Proxy-Container erkennen und eine Verbindung zu diesen herstellen.

Wenn einer Ihrer Container in der Aufgabe sich mit dem Endpunkt eines Services in einem Namespace verbinden muss, wählen Sie einen Client-Service. Wenn eine Front-End-, Reverse-Proxy- oder Load-Balancer-Anwendung externen Datenverkehr über andere Methoden empfängt, z. B. von Elastic Load Balancing, könnte sie diese Art von Service-Connect-Konfiguration verwenden.

Client-Server-Service

Ein Amazon-ECS-Service, der eine Netzwerk- oder Webservice-Anwendung ausführt. Für diesen Service müssen ein Namespace und mindestens ein Endpunkt konfiguriert sein. Jede Aufgabe im Service ist über die Endpunkte erreichbar. Der Service-Connect-Proxy-Container überwacht den Namen und Port des Endpunkts, um Datenverkehr an die App-Container in der Aufgabe weiterzuleiten.

Wenn einer der Container einen Port für Netzwerkdatenverkehr bereitstellt und überwacht, wählen Sie einen Client-Server-Service aus. Diese Anwendungen müssen keine Verbindung zu anderen Client-Server-Diensten im selben Namespace herstellen, aber die Client-Konfiguration ist erforderlich. Ein Backend, eine Middleware, eine Unternehmensebene oder die meisten Microservices können diese Art von Service Connect-Konfiguration verwenden. Wenn Sie möchten, dass eine Frontend-, Reverse Proxy- oder Load-Balancer-Anwendung Datenverkehr von anderen Services empfängt, die mit Service Connect im selben Namespace konfiguriert sind, sollten diese Services diese Art der Service-Connect-Konfiguration verwenden.

Das Service-Connect-Feature erstellt ein virtuelles Netzwerk verwandter Services. Die gleiche Service-Konfiguration kann über mehrere verschiedene Namespaces hinweg verwendet werden, um unabhängige, aber identische Anwendungssätze auszuführen. Service Connect definiert den Proxy-Container im Amazon-ECS-Service. Auf diese Weise kann dieselbe Aufgabendefinition verwendet werden, um identische Anwendungen in unterschiedlichen Namespaces mit unterschiedlichen Service-Connect-Konfigurationen auszuführen. Jede Aufgabe, die der Dienst ausführt, führt einen Proxycontainer in der Aufgabe aus.

Service Connect ist für Verbindungen zwischen Amazon-ECS-Services innerhalb desselben Namespaces geeignet. Für die folgenden Anwendungen müssen Sie eine zusätzliche Verbindungsmethode verwenden, um eine Verbindung zu einem Amazon-ECS-Service herzustellen, der mit Service Connect konfiguriert ist:

- Aufgaben, die in anderen Namespaces konfiguriert sind
- Aufgaben, die nicht für Service Connect konfiguriert sind
- Andere Anwendungen außerhalb von Amazon ECS

Diese Anwendungen können eine Verbindung über den Service-Connect-Proxy herstellen, aber keine Service-Connect-Endpunktnamen auflösen.

Damit diese Anwendungen die IP-Adressen der Amazon ECS-Aufgaben auflösen können, müssen Sie eine andere Verbindungsmethode verwenden.

Preisgestaltung

Die Preise für Amazon ECS Service Connect hängen davon ab, ob Sie Amazon EC2 EC2-Infrastruktur zum Hosten Ihrer containerisierten Workloads verwenden AWS Fargate . Wenn Sie Amazon ECS auf verwenden AWS Outposts, folgt die Preisgestaltung dem gleichen Modell wie bei der direkten Nutzung von Amazon EC2. Weitere Informationen finden Sie unter Amazon-ECS-Preise.

AWS Cloud Map Die Nutzung ist völlig kostenlos, wenn Service Connect sie verwendet.

Komponenten von Amazon ECS Service Connect

Wenn Sie Amazon ECS Service Connect verwenden, konfigurieren Sie jeden Amazon ECS-Service so, dass er eine Serveranwendung ausführt, die Netzwerkanfragen empfängt (Client-Server-Service), oder eine Client-Anwendung ausführt, die die Anfragen stellt (Client-Service).

Wenn Sie sich auf die Verwendung von Service Connect vorbereiten, beginnen Sie mit einem Client-Server-Service. Sie können eine Service-Connect-Konfiguration zu einem neuen Service oder einem vorhandenen Service hinzufügen. Amazon ECS erstellt einen Service Connect-Endpunkt im Namespace. Amazon ECS erstellt außerdem eine neue Bereitstellung im Service, um die derzeit ausgeführten Aufgaben zu ersetzen.

Vorhandene Aufgaben und andere Anwendungen können weiterhin eine Verbindung zu vorhandenen Endpunkten und externen Anwendungen herstellen. Wenn ein Client-Server-Service Aufgaben durch Skalierung hinzufügt, werden neue Verbindungen von Clients auf alle Aufgaben verteilt. Wenn ein Client-Server-Dienst aktualisiert wird, werden neue Verbindungen von Clients auf die Aufgaben der neuen Version verteilt.

Vorhandene Aufgaben können nicht aufgelöst werden und keine Verbindung zum neuen Endpunkt herstellen. Nur neue Aufgaben mit einer Service Connect-Konfiguration im selben Namespace, die nach dieser Bereitstellung ausgeführt werden, können aufgelöst und eine Verbindung zu diesem Endpunkt hergestellt werden.

Das bedeutet, dass der Betreiber der Client-Anwendung bestimmt, wann sich die Konfiguration seiner Anwendung ändert. Allerdings kann der Betreiber der Server-Anwendung seine Konfiguration jederzeit ändern. Die Liste der Endpunkte im Namespace kann sich jedes Mal ändern, wenn ein Dienst im Namespace bereitgestellt wird. Bestehende Aufgaben und Ersatzaufgaben verhalten sich weiterhin genauso wie nach der letzten Bereitstellung.

Betrachten Sie die folgenden Beispiele:

Gehen Sie zunächst davon aus, dass Sie eine Anwendung erstellen, die im öffentlichen Internet in einer einzigen AWS CloudFormation Vorlage und einem einzigen AWS CloudFormation Stapel verfügbar ist. Die öffentliche Erkennung und Erreichbarkeit sollte zuletzt erstellt werden AWS CloudFormation, einschließlich des Frontend-Client-Dienstes. Die Services müssen in dieser Reihenfolge erstellt werden, um einen Zeitraum zu verhindern, in dem der Frontend-Client-Service ausgeführt wird und öffentlich verfügbar ist, ein Backend jedoch nicht. Dadurch wird verhindert, dass während dieses Zeitraums Fehlermeldungen an die Öffentlichkeit gesendet werden. In müssen Sie das verwenden AWS CloudFormation, `dependsOn` um darauf hinzuweisen AWS CloudFormation , dass mehrere Amazon ECS-Services nicht parallel oder gleichzeitig ausgeführt werden können. Sie sollten das `dependsOn` zum Frontend-Client-Service für jeden Backend-Client-Server-Service hinzufügen, mit dem die Client-Aufgaben eine Verbindung herstellen.

Nehmen Sie zweitens an, dass ein Front-End-Service ohne Service-Connect-Konfiguration vorhanden ist. Die Aufgaben stellen eine Verbindung zu einem vorhandenen Backend-Service her. Fügen Sie dem Backend-Service zuerst eine Client-Server-Service-Connect-Konfiguration hinzu, und verwenden Sie denselben Namen im DNS oder `clientAlias`, den das Frontend verwendet. Dadurch wird eine neue Bereitstellung erstellt, d. h. alle Implementierungs-Rollback-Erkennung oder AWS SDKs und andere Methoden AWS Management Console AWS CLI, um den Backend-Service rückgängig zu machen und auf die vorherige Bereitstellung und Konfiguration zurückzusetzen. Wenn Sie mit der Leistung und dem Verhalten des Backend-Service zufrieden sind, fügen Sie dem Frontend-Service eine Client- oder Client-Server-Service-Connect-Konfiguration hinzu. Nur die Aufgaben in der neuen Bereitstellung verwenden den Service-Connect-Proxy, der diesen neuen Aufgaben hinzugefügt wurde. Wenn Sie Probleme mit dieser Konfiguration haben, können Sie ein Rollback durchführen und zu Ihrer vorherigen Konfiguration zurückkehren, indem Sie die Rollback-Erkennung für die Bereitstellung oder AWS Management Console,, AWS SDKs und andere Methoden verwenden AWS CLI, um den Back-End-Dienst auf die vorherige Bereitstellung und Konfiguration zurückzusetzen. Wenn Sie anstelle von Service Connect ein anderes Serviceerkennungssystem verwenden, das auf DNS basiert, beginnen alle Front-End- oder Client-Anwendungen mit der Verwendung neuer Endpunkte und geänderter Endpunktkonfigurationen, nachdem der lokale DNS-Cache abgelaufen ist, was normalerweise mehrere Stunden dauert.

Netzwerk

Standardmäßig überwacht der Service Connect-Proxy die Portzuweisung `containerPort` aus der Aufgabendefinition. Ihre Sicherheitsgruppenregeln müssen eingehenden (eingehenden) Datenverkehr zu diesem Port aus den Subnetzen zulassen, in denen Clients ausgeführt werden.

Auch wenn Sie in der Service-Connect-Service-Konfiguration eine Portnummer festlegen, ändert sich dadurch nicht der Port für den Client-Server-Service, den der Service-Connect-Proxy überwacht. Wenn Sie diese Portnummer festlegen, ändert Amazon ECS den Port des Endpunkts, mit dem die Client-Services eine Verbindung herstellen, auf dem Service-Connect-Proxy innerhalb dieser Aufgaben. Der Proxy im Client-Service stellt über den `containerPort` eine Verbindung mit dem Proxy im Client-Server-Service her.

Wenn Sie den Port ändern möchten, den der Service-Connect-Proxy überwacht, ändern Sie den `ingressPort0verride` in der Service-Connect-Konfiguration des Client-Server-Service. Wenn Sie diese Portnummer ändern, müssen Sie eingehenden Verkehr auf diesem Port zulassen, der für den Verkehr zu diesem Dienst verwendet wird.

Datenverkehr, den Ihre Anwendungen an Amazon-ECS-Services senden, die für Service Connect konfiguriert sind, erfordern, dass die Amazon VPC und die Subnetze über Routing-Tabellenregeln und Netzwerk-ACL-Regeln verfügen, die die von Ihnen verwendeten `containerPort`- und `ingressPort0verride`-Portnummern zulassen.

Sie können Service Connect verwenden, um Datenverkehr zwischen VPCs zu senden. Für beide VPCs gelten dieselben Anforderungen für Routingtabellen, Netzwerk-ACLs und Sicherheitsgruppen.

Beispielsweise erstellen zwei Cluster Aufgaben in verschiedenen VPCs. Ein Service in jedem Cluster ist so konfiguriert, dass er denselben Namespace verwendet. Die Anwendungen in diesen beiden Services können jeden Endpunkt im Namespace ohne VPC-DNS-Konfiguration auflösen. Die Proxys können jedoch keine Verbindung herstellen, es sei denn, die VPC-Peering-, VPC- oder Subnetz-Routing-Tabellen und VPC-Netzwerk-ACLs lassen den Verkehr auf den und Port-Nummern zu. `containerPort` `ingressPort0verride`

Für Aufgaben, die den `bridge` Netzwerkmodus verwenden, müssen Sie eine Sicherheitsgruppe mit einer Regel für eingehenden Datenverkehr erstellen, die den Datenverkehr im oberen dynamischen Portbereich zulässt. Weisen Sie dann die Sicherheitsgruppe allen EC2-Instances im Service Connect-Cluster zu.

Service-Connect-Proxy

Wenn Sie einen Service mit Service Connect-Konfiguration erstellen oder aktualisieren, fügt Amazon ECS jeder neuen Aufgabe beim Start einen neuen Container hinzu. Dieses Nutzungsmuster eines separaten Containers wird als `sidecar` bezeichnet. Dieser Container ist in der Aufgabendefinition nicht enthalten und Sie können ihn nicht konfigurieren. Amazon ECS verwaltet die Container-Konfiguration im Service. Auf diese Weise können Sie dieselben Aufgabendefinitionen zwischen mehreren Diensten, Namespaces und Aufgaben ohne Service Connect wiederverwenden.

Proxyressourcen

- Für Aufgabendefinitionen müssen Sie die CPU- und Speicherparameter festlegen.

  Wir empfehlen, die Aufgaben-CPU und den Arbeitsspeicher für den Service-Connect-Proxy-Container um 256 CPU-Einheiten und mindestens 64 MiB zu erweitern. Bei AWS Fargate ist die niedrigste Speichermenge, die Sie einstellen können, 512 MiB Speicher. In Amazon EC2 ist Speicher für Aufgabendefinitionen erforderlich.
- Für den Dienst legen Sie die Protokollkonfiguration in der Service Connect-Konfiguration fest.
- Wenn Sie erwarten, dass die Aufgaben in diesem Service bei ihrer Spitzenlast mehr als 500 Anfragen pro Sekunde erhalten, empfehlen wir, Ihre Aufgaben-CPU in dieser Aufgabendefinition für den Service-Connect-Proxy-Container um 512 CPU-Einheiten zu erweitern.
- Wenn Sie mehr als 100 Service-Connect-Services im Namespace oder insgesamt 2 000 Aufgaben für alle Amazon-ECS-Services im Namespace erstellen möchten, empfehlen wir, den Aufgabenspeicher für den Service-Connect-Proxy-Container um 128 MiB zu erweitern. Sie sollten dies in jeder Aufgabendefinition tun, die von allen Amazon-ECS-Services im Namespace verwendet wird.

Proxykonfiguration

Ihre Anwendungen verbinden sich mit dem Proxy im Beiwagen-Container in derselben Aufgabe, in der sich die Anwendung befindet. Amazon ECS konfiguriert die Aufgabe und die Container so, dass Anwendungen nur dann eine Verbindung zum Proxy herstellen, wenn die Anwendung eine Verbindung zu den Endpunktnamen im selben Namespace herstellt. Der gesamte verbleibende Datenverkehr verwendet den Proxy nicht. Der andere Datenverkehr umfasst IP-Adressen in derselben VPC, AWS Dienstendpunkte und externen Datenverkehr.

Load Balancing

Service Connect konfiguriert den Proxy so, dass er die Round-Robin-Strategie für das Load Balancing zwischen den Aufgaben in einem Service-Connect-Endpunkt verwendet. Der lokale Proxy, der sich in der Aufgabe befindet, von der die Verbindung stammt, wählt eine der Aufgaben im Client-Server-Service aus, der den Endpunkt bereitstellt.

Stellen Sie sich zum Beispiel eine Aufgabe vor, die WordPress in einem Dienst ausgeführt wird, der als Client-Dienst in einem Namespace namens local konfiguriert ist. Es gibt einen weiteren Service mit 2 Aufgaben, die die MySQL-Datenbank ausführen. Dieser Service ist so konfiguriert, dass er einen Endpunkt namens `mysql` bereitstellt, der über Service Connect im

selben Namespace aufgerufen wird. In der WordPress Aufgabe stellt die WordPress Anwendung mithilfe des Endpunktnamens eine Verbindung zur Datenbank her. Verbindungen mit diesem Namen gehen an den Proxy, der in derselben Aufgabe in einem Sidecar-Container ausgeführt wird. Anschließend kann der Proxy mithilfe der Round-Robin-Strategie eine Verbindung zu einer der MySQL-Aufgaben herstellen.

Strategien für das Load Balancing: Round-Robin

Ausreißererkennung

Dieses Feature verwendet Daten, die dem Proxy über frühere fehlgeschlagene Verbindungen vorliegen, um zu verhindern, dass neue Verbindungen an die Hosts gesendet werden, bei denen Verbindungen fehlgeschlagen sind. Service Connect konfiguriert das Feature zur Erkennung von Ausreißern des Proxys für passive Zustandsprüfungen.

Im vorherigen Beispiel kann der Proxy eine Verbindung zu einer der MySQL-Aufgaben herstellen. Wenn der Proxy mehrere Verbindungen zu einer bestimmten MySQL-Aufgabe hergestellt hat und 5 oder mehr der Verbindungen in den letzten 30 Sekunden fehlgeschlagen sind, vermeidet der Proxy diese MySQL-Aufgabe für 30 bis 300 Sekunden.

Wiederholversuche

Service Connect konfiguriert den Proxy so, dass Verbindungen, die den Proxy passieren und fehlschlagen, erneut versucht werden, und beim zweiten Versuch wird vermieden, den Host der vorherigen Verbindung zu verwenden. Dadurch wird sichergestellt, dass jede Verbindung über Service Connect nicht aus einmaligen Gründen fehlschlägt.

Anzahl der Wiederholungen: 2

Zeitüberschreitung

Service Connect konfiguriert den Proxy so, dass er eine maximale Zeitspanne auf die Antwort Ihrer Client-Server-Anwendungen wartet. Der Standardwert für das Timeout beträgt 15 Sekunden, kann aber aktualisiert werden.

Optionale Parameter:

idleTimeout — Die Zeit in Sekunden, für die eine Verbindung aktiv bleibt, wenn sie inaktiv ist. Der Wert deaktiviert. `0` `idleTimeout`

Die `idleTimeout` Standardeinstellung für`HTTP/HTTP2/GRPC`ist 5 Minuten.

Die `idleTimeout` Standardeinstellung für `TCP` ist 1 Stunde.

pro RequestTimeout - Die Wartezeit, bis der Upstream mit einer vollständigen Antwort pro Anfrage antwortet. Der Wert `0` schaltet sich aus`perRequestTimeout`. Dies kann nur festgelegt werden, wenn der `appProtocol` For-Anwendungscontainer `HTTPHTTP2/istGRPC`. Die Standardeinstellung ist 15 Sekunden.

> **Note**
>
> Wenn auf eine Zeit eingestellt `idleTimeout` ist, die kürzer als ist`perRequestTimeout`, wird die Verbindung geschlossen, wenn der erreicht `idleTimeout` ist und nicht der`perRequestTimeout`.

Überlegungen

Beachten Sie bei der Verwendung von Service Connect Folgendes:

- Aufgaben, die in Fargate ausgeführt werden, müssen die Fargate Linux-Plattformversion `1.4.0` oder höher verwenden, um Service Connect verwenden zu können.
- Die Amazon ECS-Agentenversion auf der Container-Instance muss `1.67.2` oder höher sein.
- Container Instances müssen die Amazon-ECS-optimierte AMI-Version für Amazon Linux 2023 `20230428` oder höher oder die Amazon-ECS-optimierte AMI-Version für Amazon Linux 2 `2.0.20221115` ausführen, um Service Connect nutzen zu können. Diese Versionen verfügen zusätzlich zum Amazon-ECS-Container-Agenten über den Service-Connect-Agenten. Weitere Informationen zum Service Connect-Agenten finden Sie unter Amazon ECS Service Connect Agent unter GitHub.
- Container-Instances müssen über die `ecs:Poll`-Berechtigung für die Ressource `arn:aws:ecs:`*`region`*`:`*`0123456789012`*`:task-set/`*`cluster`*`/*` verfügen. Wenn Sie die `ecsInstanceRole` verwenden, müssen Sie keine zusätzlichen Berechtigungen hinzufügen. Die `AmazonEC2ContainerServiceforEC2Role`-verwaltete Richtlinie verfügt über die erforderlichen Berechtigungen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.
- Nur Services, die fortlaufende Bereitstellungen verwenden, werden in Service Connect unterstützt.
- Aufgaben, die den `bridge` Netzwerkmodus verwenden und Service Connect verwenden, unterstützen den `hostname` Container-Definitionsparameter nicht.
- Aufgabendefinitionen müssen das zu verwendende Speicherlimit für Service Connect festlegen. Weitere Informationen finden Sie unter Service-Connect-Proxy.

- Aufgabendefinitionen, die Speicherlimits für Container festlegen, werden nicht unterstützt.

  Sie können Container-Speicherlimits für Ihre Container festlegen, aber Sie müssen das Aufgaben-Speicherlimit auf eine Zahl festlegen, die größer ist als die Summe der Container-Speicherlimits. Die zusätzliche CPU und der zusätzliche Arbeitsspeicher in den Aufgabenlimits, die nicht in den Containerlimits zugewiesen sind, werden vom Service-Connect-Proxy-Container und anderen Containern verwendet, die keine Containerlimits festlegen. Weitere Informationen finden Sie unter Service-Connect-Proxy.
- Sie können Service Connect so konfigurieren, dass ein beliebiger AWS Cloud Map Namespace in derselben Region in derselben AWS-Konto verwendet wird.
- Jeder Dienst kann nur zu einem Namespace gehören.
- Nur die Aufgaben, die Dienste erstellen, werden unterstützt.
- Alle Endpunkte müssen innerhalb eines Namespace eindeutig sein.
- Alle Erkennungsnamen müssen in einem Namespace eindeutig sein.
- Sie müssen vorhandene Dienste erneut bereitstellen, bevor die Anwendungen neue Endpunkte auflösen können. Neue Endpunkte, die dem Namespace nach der letzten Bereitstellung hinzugefügt werden, werden nicht zur Aufgabenkonfiguration hinzugefügt. Weitere Informationen finden Sie unter the section called “Service Connect-Komponenten”.
- Service Connect löscht keine Namespaces, wenn Cluster gelöscht werden. Sie müssen Namespaces in löschen. AWS Cloud Map
- Der Application Load Balancer Balancer-Datenverkehr wird standardmäßig im `awsvpc` Netzwerkmodus über den Service Connect-Agenten geleitet. Wenn Sie möchten, dass dienstfreier Datenverkehr den Service Connect-Agenten umgeht, verwenden Sie den `ingressPortOverride` Parameter in Ihrer Service Connect-Dienstkonfiguration.

Service Connect unterstützt Folgendes nicht:

- Windows-Container
- HTTP 1.0
- Eigenständige Aufgaben
- Dienste, die die Bereitstellungstypen Blau/Grün und externe Bereitstellungen verwenden
- `External`-Container-Instances für Amazon ECS Anywhere werden nicht in Service Connect unterstützt.
- PPv2

Regionen mit Service Connect

Amazon ECS Service Connect ist in den folgenden AWS Regionen verfügbar:

Name der Region	Region
USA Ost (Ohio)	us-east-2
USA Ost (Nord-Virginia)	us-east-1
USA West (Nordkalifornien)	us-west-1
USA West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Hyderabad)	ap-south-2
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Melbourne)	ap-southeast-4
Asien-Pazifik (Tokio)	ap-northeast-1
Kanada (Zentral)	ca-central-1
Kanada West (Calgary)	ca-west-1
China (Peking)	cn-north-1 (Hinweis: TLS für Service Connect ist in dieser Region nicht verfügbar.)

Name der Region	Region
China (Ningxia)	cn-northwest-1 (Hinweis: TLS für Service Connect ist in dieser Region nicht verfügbar.)
Europa (Frankfurt)	eu-central-1
Europa (Irland)	eu-west-1
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Mailand)	eu-south-1
Europa (Spanien)	eu-south-2
Europa (Stockholm)	eu-north-1
Europa (Zürich)	eu-central-2
Israel (Tel Aviv)	il-central-1
Naher Osten (Bahrain)	me-south-1
Naher Osten (VAE)	me-central-1
Südamerika (São Paulo)	sa-east-1

Überblick über die Konfiguration von Amazon ECS Service Connect

Wenn Sie Service Connect verwenden, müssen Sie Parameter in Ihren Ressourcen konfigurieren.

Amazon ECS-Ressourcen, die für Service Connect konfiguriert werden müssen

Parameter-Speicherort	Anwendungstyp	Beschreibung	Erforderlich
Aufgabendefinition	Client	Für Service Connect in Client-Aufgabendefinitionen sind keine Änderungen verfügbar.	N/A

Parameter-Speicherort	Anwendungstyp	Beschreibung	Erforderlich
Aufgabendefinition	Client-Server	Server müssen `name`-Felder zu Ports in den `portMappings` von Containern hinzufügen. Weitere Informationen finden Sie unter `portMappings`.	Ja
Aufgabendefinition	Client-Server	Server können optional ein Anwendungsprotokoll (z. B. HTTP) bereitstellen, um protokollspezifische Metriken für ihre Serveranwendungen zu erhalten (z. B. `HTTP 5xx`).	Nein
Service-Definition	Client	Client-Services müssen eine `serviceConnectConfiguration` hinzufügen, um den beizutretenden Namespace zu konfigurieren. Dieser Namespace muss alle Server-Services enthalten, die dieser Service erkennen muss. Weitere Informationen finden Sie unter `serviceConnectConfiguration`.	Ja
Service-Definition	Client-Server	Server-Services müssen eine `serviceConnectConfiguration` hinzufügen, um die DNS-Namen, Portnummern und den Namespace zu konfigurieren, über den der Service verfügbar ist. Weitere Informationen finden Sie unter `serviceConnectConfiguration`.	Ja

Parameter-Speicherort	Anwendungstyp	Beschreibung	Erforderlich
Cluster	Client	Cluster können einen standardmäßigen Service-Connect-Namespace hinzufügen. Neue Services im Cluster erben den Namespace, wenn Service Connect in einem Service konfiguriert ist.	Nein
Cluster	Client-Server	Für Service Connect in Clustern gibt es keine Änderungen, die sich auf Server-Services beziehen. Server-Aufgabendefinitionen und -Services müssen die entsprechende Konfiguration festlegen.	N/A

Übersicht der Schritte zum Konfigurieren von Service Connect

Die folgenden Schritte bieten einen Überblick über die Konfiguration von Service Connect.

Important

- Service Connect erstellt AWS Cloud Map Dienste in Ihrem Konto. Wenn Sie diese AWS Cloud Map -Ressourcen ändern, indem Sie Instances manuell registrieren/deregistrieren, Instance-Attribute ändern oder einen Service löschen, kann dies zu unerwartetem Verhalten bei Ihrem Anwendungsdatenverkehr oder nachfolgenden Bereitstellungen führen.
- Service Connect unterstützt keine Links in der Aufgabendefinition.

1. Fügen Sie Portnamen zu den Portzuordnungen in Ihren Aufgabendefinitionen hinzu. Außerdem können Sie das Layer-7-Protokoll der Anwendung identifizieren, um zusätzliche Metriken zu erhalten.
2. Erstellen Sie einen Cluster mit einem AWS Cloud Map Namespace oder erstellen Sie den Namespace separat. Für eine einfache Organisation erstellen Sie einen Cluster mit dem Namen, den Sie für den Namespace wünschen, und geben Sie den identischen Namen für den Namespace an. In diesem Fall erstellt Amazon ECS einen neuen HTTP-Namespace mit der

erforderlichen Konfiguration. Service Connect verwendet oder erstellt keine DNS-gehosteten Zonen in Amazon Route 53.

3. Konfigurieren Sie Services, um Service-Connect-Endpunkte innerhalb des Namespace zu erstellen.
4. Stellen Sie Services bereit, um die Endpunkte zu erstellen. Amazon ECS fügt jeder Aufgabe einen Service-Connect-Proxy-Container hinzu und erstellt die Service-Connect-Endpunkte in AWS Cloud Map. Dieser Container ist in der Aufgabendefinition nicht konfiguriert, und die Aufgabendefinition kann ohne Änderung wiederverwendet werden, um mehrere Services im selben Namespace oder in mehreren Namespaces zu erstellen.
5. Stellen Sie Client-Anwendungen als Services bereit, um eine Verbindung zu den Endpunkten herzustellen. Amazon ECS verbindet diese mit den Service-Connect-Endpunkten über den Service-Connect-Proxy in jeder Aufgabe.

   Anwendungen verwenden den Proxy nur, um sich mit Service-Connect-Endpunkten zu verbinden. Es gibt keine zusätzliche Konfiguration für die Verwendung des Proxys. Der Proxy führt Round-Robin-Load-Balancing, die Erkennung von Ausreißern und Wiederholungsversuche durch. Weitere Informationen zum Proxy finden Sie unter Service-Connect-Proxy.
6. Überwachen Sie den Verkehr über den Service Connect-Proxy in Amazon CloudWatch.

Cluster-Konfiguration

Sie können einen Standardnamespace für Service Connect festlegen, wenn Sie den Cluster erstellen oder aktualisieren. Wenn Sie einen Namespace-Namen angeben, der nicht in derselben AWS-Region und demselben Konto vorhanden ist, wird ein neuer HTTP-Namespace erstellt.

Wenn Sie einen Cluster erstellen und einen Standard-Service-Connect-Namespace angeben, wartet der Cluster im `PROVISIONING`-Status, während Amazon ECS den Namespace erstellt. Im Status des Clusters sehen Sie einen `attachment`, der den Status des Namespace anzeigt. Anlagen werden standardmäßig nicht in der angezeigt. Sie müssen sie hinzufügen AWS CLI, um sie `--include ATTACHMENTS` zu sehen.

Konfiguration des Dienstes

Service Connect ist so konzipiert, dass nur ein Minimum an Konfiguration erforderlich ist. Sie müssen für jede Portzuordnung, die Sie mit Service Connect verwenden möchten, in der Aufgabendefinition einen Namen festlegen. Im Service müssen Sie Service Connect aktivieren und einen Namespace auswählen, um einen Client-Service zu erstellen. Um einen Client-Server-Service zu erstellen,

müssen Sie eine einzelne Service-Connect-Service-Konfiguration hinzufügen, die dem Namen einer der Portzuordnungen entspricht. Amazon ECS verwendet die Portnummer und den Portnamen aus der Aufgabendefinition erneut, um den Service-Connect-Service und -Endpunkt zu definieren. Um diese Werte zu überschreiben, können Sie die anderen Parameter Discovery (Erkennung), DNS und Port in der Konsole oder `discoveryName` bzw. `clientAliases` in der Amazon-ECS-API verwenden.

Das folgende Beispiel zeigt, wie jede Art von Service-Connect-Konfiguration zusammen im selben Amazon-ECS-Service verwendet wird. Shell-Kommentare werden bereitgestellt. Berücksichtigen Sie jedoch, dass die für Amazon-ECS-Services verwendete JSON-Konfiguration keine Kommentare unterstützt.

```
{
    ...
    serviceConnectConfiguration: {
        enabled: true,
        namespace: "internal",
        #config for client services can end here, only these two parameters are
 required.
        services: [{
                portName: "http"
            }, #minimal client - server service config can end here.portName must match
 the "name"
            parameter of a port mapping in the task definition. {
                discoveryName: "http-second"
                #name the discoveryName to avoid a Task def port name collision with
 the minimal config in the same Cloud Map namespace
                portName: "http"
            },
            {
                clientAliases: [{
                        dnsName: "db",
                        port: 81
                    }] #use when the port in Task def is not the port that client apps
 use.Client apps can use http: //db:81 to connect
                    discoveryName: "http-three"
                portName: "http"
            },
            {
                clientAliases: [{
                    dnsName: "db.app",
```

```
                    port: 81
                }] #use when the port in Task def is not the port that client apps
 use.duplicates are fine as long as the discoveryName is different.
                discoveryName: "http-four"
                portName: "http",
                ingressPortOverride: 99 #If App should also accept traffic directly on
 Task def port.
            }
        ]
    }
}
```

Verschlüsseln Sie den Amazon ECS Service Connect-Verkehr

Amazon ECS Service Connect unterstützt die automatische Verschlüsselung des Datenverkehrs mit Transport Layer Security (TLS) -Zertifikaten für Amazon ECS-Services. Wenn Sie Ihre Amazon ECS-Services auf ein AWS Private Certificate Authority (AWS Private CA) verweisen, stellt Amazon ECS automatisch TLS-Zertifikate zur Verschlüsselung des Datenverkehrs zwischen Ihren Amazon ECS Service Connect-Services bereit. Amazon ECS generiert, rotiert und verteilt TLS-Zertifikate, die für die Verschlüsselung des Datenverkehrs verwendet werden.

Die automatische Datenverkehrsverschlüsselung mit Service Connect nutzt branchenführende Verschlüsselungsfunktionen, um Ihre dienstübergreifende Kommunikation zu sichern, sodass Sie Ihre Sicherheitsanforderungen erfüllen können. Es unterstützt AWS Private Certificate Authority TLS-Zertifikate mit `256-bit ECDSA` und `2048-bit RSA` Verschlüsselung. Standardmäßig wird TLS 1.3 unterstützt, TLS 1.0 — 1.2 werden jedoch nicht unterstützt. Sie haben auch die volle Kontrolle über private Zertifikate und Signaturschlüssel, sodass Sie die Compliance-Anforderungen erfüllen können.

> **Note**
>
> Um TLS 1.3 verwenden zu können, müssen Sie es auf dem Listener auf dem Ziel aktivieren. Nur eingehender und ausgehender Datenverkehr, der den Amazon ECS-Agenten passiert, ist verschlüsselt.

AWS Private Certificate Authority Zertifikate und Service Connect

Für die Ausstellung von Zertifikaten sind zusätzliche IAM-Berechtigungen erforderlich. Amazon ECS bietet eine Vertrauensrichtlinie für verwaltete Ressourcen, in der der Satz von Berechtigungen

beschrieben wird. Weitere Informationen zu dieser Richtlinie finden Sie unter AmazonECS InfrastructureRole PolicyFor ServiceConnect TransportLayer Security.

AWS Private Certificate Authority Modi für Service Connect

AWS Private Certificate Authority kann in zwei Modi ausgeführt werden: allgemein und kurzlebig.

- Universell einsetzbar — Zertifikate, die mit einem beliebigen Ablaufdatum konfiguriert werden können.
- Kurzlebig - Zertifikate mit einer maximalen Gültigkeitsdauer von sieben Tagen.

Amazon ECS unterstützt zwar beide Modi, wir empfehlen jedoch, kurzlebige Zertifikate zu verwenden. Standardmäßig rotieren Zertifikate alle fünf Tage, und der Betrieb im kurzlebigen Modus bietet erhebliche Kosteneinsparungen im Vergleich zu herkömmlichen Zertifikaten.

Service Connect unterstützt den Widerruf von Zertifikaten nicht und nutzt stattdessen kurzlebige Zertifikate mit häufiger Zertifikatsrotation. Sie sind berechtigt, die Rotationsfrequenz zu ändern, die Geheimnisse mithilfe der verwalteten Rotation im Secrets Manager zu deaktivieren oder zu löschen. Dies kann jedoch die folgenden möglichen Konsequenzen haben.

- Kürzere Rotationsfrequenz - Eine kürzere Rotationsfrequenz verursacht höhere Kosten AWS Private CA, da Secrets Manager AWS KMS und Auto Scaling einen erhöhten Arbeitsaufwand für die Rotation aufweisen.
- Längere Rotationsfrequenz - Die Kommunikation Ihrer Anwendungen schlägt fehl, wenn die Rotationsfrequenz sieben Tage überschreitet.
- Löschung des Geheimnisses — Das Löschen des Geheimnisses führt zu einem Fehler bei der Rotation und beeinträchtigt die Kommunikation mit den Kundenanwendungen.

Falls Ihre geheime Rotation fehlschlägt, wird ein `RotationFailed` Ereignis unter veröffentlicht AWS CloudTrail. Sie können auch einen CloudWatch Alarm für einrichten`RotationFailed`.

> **Important**
>
> Fügen Sie keine Replikatregionen zu Geheimnissen hinzu. Dadurch wird verhindert, dass Amazon ECS das Geheimnis löscht, da Amazon ECS nicht berechtigt ist, Regionen aus der Replikation zu entfernen. Wenn Sie die Replizierung bereits hinzugefügt haben, führen Sie den folgenden Befehl aus.

```
aws secretsmanager remove-regions-from-replication \
 --secret-id SecretId \
 --remove-replica-regions region-name
```

Untergeordnete Zertifizierungsstellen

Sie können alle AWS Private CA, ob Stamm- oder untergeordnetes System, zu Service Connect TLS hinzufügen, um Endentitätszertifikate für die Dienste auszustellen. Der angegebene Emittent wird überall als Unterzeichner und Vertrauensanker behandelt. Sie können Endzertifikate für verschiedene Teile Ihrer Anwendung von verschiedenen untergeordneten Zertifizierungsstellen ausstellen. Wenn Sie den verwenden AWS CLI, geben Sie den Amazon-Ressourcennamen (ARN) der Zertifizierungsstelle an, um die Vertrauenskette einzurichten.

Lokale Zertifizierungsstellen

Um Ihre lokale Zertifizierungsstelle zu verwenden, erstellen und konfigurieren Sie eine untergeordnete Zertifizierungsstelle in. AWS Private Certificate Authority Dadurch wird sichergestellt, dass alle TLS-Zertifikate, die für Ihre Amazon ECS-Workloads ausgestellt wurden, die Vertrauenskette mit den Workloads teilen, die Sie vor Ort ausführen, und dass sie eine sichere Verbindung herstellen können.

> **Important**
>
> Fügen Sie das erforderliche Tag `AmazonECSManaged : true` zu Ihrem hinzu. AWS Private CA

Infrastructure as Code

Wenn Sie Service Connect TLS mit Infrastructure as Code (IaC) -Tools verwenden, ist es wichtig, dass Sie Ihre Abhängigkeiten korrekt konfigurieren, um Probleme zu vermeiden, z. B. wenn Dienste nicht mehr ausgeführt werden. Ihr AWS KMS Schlüssel, falls angegeben, Ihre IAM-Rolle und Ihre AWS Private CA Abhängigkeiten sollten nach Ihrem Amazon ECS-Service gelöscht werden.

Service Connect und AWS Key Management Service

Sie können AWS Key Management Servicedamit Ihre Service Connect-Ressourcen ver- und entschlüsseln. AWS KMS ist ein von ihm verwalteter Dienst AWS , mit dem Sie kryptografische Schlüssel zum Schutz Ihrer Daten erstellen und verwalten können.

Bei der Verwendung AWS KMS mit Service Connect können Sie entweder einen AWS eigenen Schlüssel verwenden, der für Sie AWS verwaltet wird, oder Sie können einen vorhandenen AWS KMS Schlüssel wählen. Sie können auch einen neuen AWS KMS Schlüssel zur Verwendung erstellen.

Bereitstellung Ihres eigenen Verschlüsselungsschlüssels

Sie können Ihre eigenen Schlüsselmaterialien bereitstellen oder einen externen Schlüsselspeicher verwenden, indem Sie Ihren eigenen Schlüssel AWS Key Management Service importieren in AWS KMS verwenden und dann den Amazon-Ressourcennamen (ARN) dieses Schlüssels in Amazon ECS Service Connect angeben.

Im Folgenden finden Sie ein Beispiel AWS KMS für eine Richtlinie. Ersetzen Sie die *Benutzereingabewerte* durch Ihre eigenen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "id",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role-name"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyPair"
      ],
      "Resource": "*"
    }
  ]
}
```

Weitere Informationen zu wichtigen Richtlinien finden Sie unter Erstellen einer Schlüsselrichtlinie im AWS Key Management Service Entwicklerhandbuch.

> **Note**
>
> Service Connect unterstützt nur symmetrische AWS KMS Verschlüsselungsschlüssel. Sie können keine andere Art von AWS KMS Schlüssel verwenden, um Ihre Service Connect-Ressourcen zu verschlüsseln. Hilfe bei der Bestimmung, ob es sich bei einem AWS KMS Schlüssel um einen symmetrischen Verschlüsselungsschlüssel handelt, finden Sie unter Identifizieren symmetrischer und asymmetrischer Schlüssel. AWS KMS

Weitere Informationen zu symmetrischen Verschlüsselungsschlüsseln finden Sie unter AWS Key Management Service Symmetrische AWS KMS Verschlüsselungsschlüssel im Entwicklerhandbuch.AWS Key Management Service

TLS für Amazon ECS Service Connect aktivieren

Sie aktivieren die Datenverkehrsverschlüsselung, wenn Sie einen Service Connect-Dienst erstellen oder aktualisieren.

Um die Verkehrsverschlüsselung für einen Dienst in einem vorhandenen Namespace zu aktivieren, verwenden Sie AWS Management Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Namespaces aus.
3. Wählen Sie den Namespace mit dem Dienst aus, für den Sie die Verkehrsverschlüsselung aktivieren möchten.
4. Wählen Sie den Dienst aus, für den Sie die Verkehrsverschlüsselung aktivieren möchten.
5. Wählen Sie in der oberen rechten Ecke Service aktualisieren und scrollen Sie nach unten zum Abschnitt Service Connect.
6. Wählen Sie unter Ihren Dienstinformationen die Option Verkehrsverschlüsselung aktivieren aus, um TLS zu aktivieren.
7. Wählen Sie für die TLS-Rolle Service Connect eine vorhandene Rolle aus, oder erstellen Sie eine neue.
8. Wählen Sie für Signer Certificate Authority eine bestehende Zertifizierungsstelle aus oder erstellen Sie eine neue.

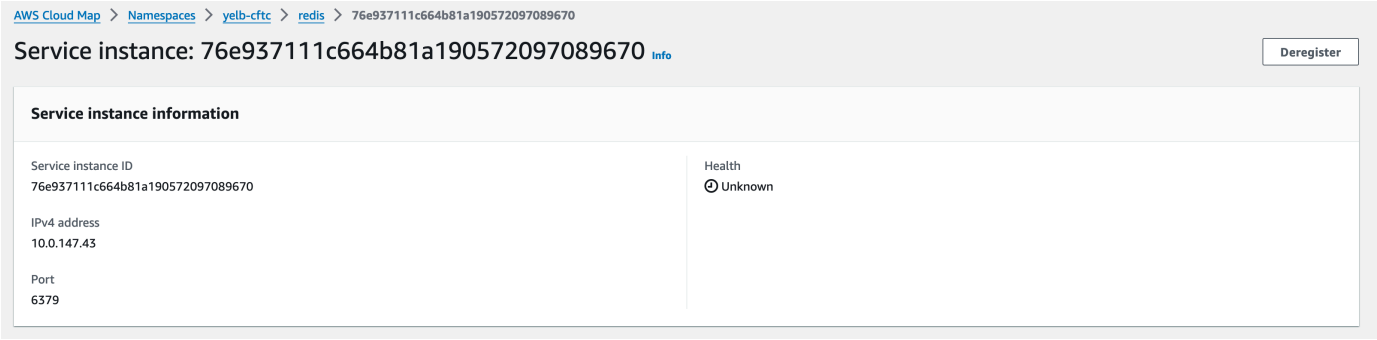
9. Wählen Sie unter Wählen Sie einen AWS eigenen und verwalteten Schlüssel aus, oder Sie können einen anderen Schlüssel wählen. AWS KMS key Sie können sich auch dafür entscheiden, einen neuen zu erstellen.

Ein Beispiel für die Verwendung von AWS CLI zur Konfiguration von TLS für Ihren Dienst finden Sie unterKonfiguration von Amazon ECS Service Connect mit dem AWS CLI.

Überprüfen, ob TLS für Amazon ECS Service Connect aktiviert ist

Service Connect initiiert TLS auf dem Service Connect-Agenten und beendet es auf dem Zielagenten. Infolgedessen sieht der Anwendungscode niemals TLS-Interaktionen. Gehen Sie wie folgt vor, um zu überprüfen, ob TLS aktiviert ist.

1. Stellen Sie sicher, dass Ihr Anwendungs-Image über die `openssl` CLI verfügt.
2. Aktivieren Sie ECS Exec auf Ihren Diensten, um über SSM eine Verbindung zu Ihren Aufgaben herzustellen. Alternativ können Sie eine Amazon EC2 EC2-Instance in derselben Amazon VPC wie der Service starten.
3. Rufen Sie die IP und den Port einer Aufgabe von einem Service ab, den Sie verifizieren möchten. Wenn für Ihren `redis` Dienst beispielsweise TLS aktiviert ist, können Sie die zugehörige Task-IP abrufen, indem Sie zu dem Dienst navigieren AWS Cloud Map, ihn suchen und sich die IP und den Port einer Instanz ansehen.



4. Melden Sie sich `execute-command` wie im folgenden Beispiel bei einer Ihrer Aufgaben an. Melden Sie sich alternativ bei der in Schritt 2 erstellten Amazon EC2-Instance an.

```
$ aws ecs execute-command --cluster cluster-name \
    --task < TASK_ID>  \
    --container app \
    --interactive \
    --command "/bin/sh"
```

> **Note**
>
> Wenn Sie den DNS-Namen direkt aufrufen, wird das Zertifikat nicht angezeigt.

5. Verwenden Sie in der verbundenen Shell die `openssl` CLI, um das an die Aufgabe angehängte Zertifikat zu überprüfen und anzuzeigen.

   Beispiel:

```
openssl s_client -connect 10.0.147.43:6379 < /dev/null 2> /dev/null \
| openssl x509 -noout -text
```

   Beispielantwort:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            <serial-number>
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: <issuer>
        Validity
            Not Before: Jan 23 21:38:12 2024 GMT
            Not After : Jan 30 22:38:12 2024 GMT
        Subject: <subject>
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    <pub>
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:redis.yelb-cftc
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Authority Key Identifier:
                keyid:<key-id>

            X509v3 Subject Key Identifier:
```

```
                1D:<id>
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
    Signature Algorithm: ecdsa-with-SHA256
        <hash>
```

Konfiguration von Amazon ECS Service Connect mit dem AWS CLI

Sie können einen Amazon ECS-Service für eine Fargate-Aufgabe erstellen, die Service Connect mit dem AWS CLI verwendet.

Voraussetzungen

Im Folgenden sind die Voraussetzungen für Service Connect aufgeführt:

- Stellen Sie sicher, dass die Region Service Connect unterstützt. Weitere Informationen finden Sie unter Regions with Service Connect.
- Stellen Sie sicher, dass die neueste Version von installiert und konfiguriert AWS CLI ist. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben eine VPC, ein Subnetz, eine Routing-Tabelle und eine Sicherheitsgruppe zur Verwendung erstellt. Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.
- Sie verfügen über eine Aufgabenausführungsrolle mit dem Namen `ecsTaskExecutionRole` und die von `AmazonECSTaskExecutionRolePolicy` verwaltete Richtlinie ist der Rolle angefügt. Diese Rolle ermöglicht es Fargate, die NGINX-Anwendungsprotokolle und Service Connect-Proxyprotokolle in Amazon Logs zu schreiben. CloudWatch Weitere Informationen finden Sie unter Erstellen der -Aufgabenausführungsrolle.

Schritt 1: Cluster erstellen

Führen Sie die folgenden Schritte aus, um Ihren Amazon-ECS-Cluster und -Namespace zu erstellen.

Um den Amazon ECS-Cluster und den AWS Cloud Map Namespace zu erstellen

1. Erstellen Sie einen zu verwendenden Amazon-ECS-Cluster mit dem Namen `tutorial`. Der Parameter `--service-connect-defaults` legt den Standard-Namespace des Clusters fest. In der Beispielausgabe ist in diesem Konto `service-connect` kein AWS Cloud Map Namespace mit dem Namen vorhanden AWS-Region, weshalb der Namespace von Amazon ECS erstellt wurde. Der Namespace wird in AWS Cloud Map im Konto erstellt und ist mit allen anderen Namespaces sichtbar. Verwenden Sie daher einen Namen, der den Zweck angibt.

   ```
   aws ecs create-cluster --cluster-name tutorial --service-connect-defaults
    namespace=service-connect
   ```

   Ausgabe:

   ```
   {
       "cluster": {
           "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
           "clusterName": "tutorial",
           "serviceConnectDefaults": {
               "namespace": "arn:aws:servicediscovery:us-
   west-2:123456789012:namespace/ns-EXAMPLE"
           },
           "status": "PROVISIONING",
           "registeredContainerInstancesCount": 0,
           "runningTasksCount": 0,
           "pendingTasksCount": 0,
           "activeServicesCount": 0,
           "statistics": [],
           "tags": [],
           "settings": [
               {
                   "name": "containerInsights",
                   "value": "disabled"
               }
           ],
           "capacityProviders": [],
           "defaultCapacityProviderStrategy": [],
           "attachments": [
               {
                   "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                   "type": "sc",
                   "status": "ATTACHING",
   ```

```
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
}
```

2. Stellen Sie sicher, dass der Cluster erstellt wurde:

```
aws ecs describe-clusters --clusters tutorial
```

Ausgabe:

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
            "clusterName": "tutorial",
            "serviceConnectDefaults": {
                "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
            },
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
    ],
    "failures": []
}
```

3. (Optional) Stellen Sie sicher, dass der Namespace in erstellt wurde. AWS Cloud Map Sie können die AWS Management Console oder die normale AWS CLI Konfiguration verwenden, in AWS Cloud Map der dieser erstellt wurde.

Verwenden Sie beispielsweise die AWS CLI:

```
aws servicediscovery --region us-west-2 get-namespace --id ns-EXAMPLE
```

Ausgabe:

```
{
    "Namespace": {
        "Id": "ns-EXAMPLE",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
EXAMPLE",
        "Name": "service-connect",
        "Type": "HTTP",
        "Properties": {
            "DnsProperties": {
                "SOA": {}
            },
            "HttpProperties": {
                "HttpName": "service-connect"
            }
        },
        "CreateDate": 1661749852.422,
        "CreatorRequestId": "service-connect"
    }
}
```

Schritt 2: Erstellen Sie den Dienst für den Server

Das Service-Connect-Feature dient zum Verbinden mehrerer Anwendungen auf Amazon ECS. Mindestens eine dieser Anwendungen muss einen Web-Service bereitstellen, zu dem eine Verbindung hergestellt werden kann. In diesem Schritt erstellen Sie:

- Die Aufgabendefinition, die das unveränderte offizielle Nginx-Container-Image verwendet und die Service-Connect-Konfiguration beinhaltet.
- Die Amazon ECS-Servicedefinition, die Service Connect so konfiguriert, dass Service Discovery und Service Mesh-Proxy für den Datenverkehr zu diesem Service bereitgestellt werden. Die Konfiguration verwendet den Standard-Namespace aus der Cluster-Konfiguration erneut, um den Umfang der Service-Konfiguration, die Sie für jeden Service vornehmen, zu reduzieren.

- Der Amazon-ECS-Service. Er führt eine Aufgabe mithilfe der Aufgabendefinition aus und fügt einen zusätzlichen Container für den Service-Connect-Proxy ein. Der Proxy überwacht den Port aus der Container-Port-Zuordnung der Aufgabendefinition. In einer Client-Anwendung, die in Amazon ECS ausgeführt wird, wartet der Proxy in der Client-Aufgabe auf ausgehende Verbindungen mit dem Portnamen der Aufgabendefinition, dem Serviceeerkennungsnamen oder dem Service-Client-Aliasnamen und der Portnummer des Client-Alias.

Um den Webservice mit Service Connect zu erstellen

1. Registrieren Sie eine Aufgabendefinition, die mit Fargate kompatibel ist und den `awsvpc`-Netzwerkmodus verwendet. Dazu gehen Sie wie folgt vor:

   a. Erstellen Sie eine Datei mit dem Namen `service-connect-nginx.json` mit dem Inhalt der folgenden Aufgabendefinition.

      Diese Aufgabendefinition konfiguriert Service Connect durch Hinzufügen von `name`- und `appProtocol`-Parametern zur Portzuordnung. Durch den Portnamen wird dieser Port in der Service-Konfiguration besser identifizierbar, wenn mehrere Ports verwendet werden. Der Portname wird standardmäßig auch als auffindbarer Name für andere Anwendungen im Namespace verwendet.

      Die Aufgabendefinition enthält die Aufgaben-IAM-Rolle, da für den Service ECS Exec aktiviert ist.

      > **Important**
      >
      > Diese Aufgabendefinition verwendet `alogConfiguration`, um die Nginx-Ausgabe von `stdout` und an Amazon CloudWatch Logs `stderr` zu senden. Diese Rolle zur Aufgabenausführung verfügt nicht über die zusätzlichen Berechtigungen, die für die Erstellung der Protokollgruppe CloudWatch Logs erforderlich sind. Erstellen Sie die Protokollgruppe in CloudWatch Logs mithilfe von AWS Management Console oder AWS CLI. Wenn Sie die Nginx-Protokolle nicht an Logs senden möchten, CloudWatch können Sie die entfernen. `logConfiguration`
      > Ersetzen Sie die AWS-Konto ID in der Rolle zur Aufgabenausführung durch Ihre AWS-Konto ID.

```
{
    "family": "service-connect-nginx",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
        "name": "webserver",
        "image": "public.ecr.aws/docker/library/nginx:latest",
        "cpu": 100,
        "portMappings": [
            {
                "name": "nginx",
                "containerPort": 80,
                "protocol": "tcp",
                "appProtocol": "http"
            }
        ],
        "essential": true,
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-nginx",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "nginx"
            }
        }
        }
    ],
    "cpu": "256",
    "memory": "512"
}
```

b. Registrieren Sie die Aufgabendefinition mit der `service-connect-nginx.json`-Datei:

```
aws ecs register-task-definition --cli-input-json file://service-connect-
nginx.json
```

2. Erstellen Sie einen Dienst:

a. Erstellen Sie eine Datei mit dem Namen `service-connect-nginx-service.json` mit dem Inhalt des Amazon-ECS-Service, den Sie erstellen. Dieses Beispiel verwendet die Aufgabendefinition, die im vorherigen Schritt erstellt wurde. Ein `awsvpcConfiguration` ist erforderlich, da die Beispiel-Aufgabendefinition den `awsvpc`-Netzwerkmodus verwendet.

   Wenn Sie den ECS-Service erstellen, geben Sie den Fargate-Starttyp und die `LATEST`-Plattformversion an, die Service Connect unterstützt. Die `securityGroups` und `subnets` müssen zu einer VPC gehören, die die Anforderungen für die Verwendung von Amazon ECS erfüllt. Sie können die Sicherheitsgruppen- und Subnetz-IDs über die Amazon-VPC-Konsole abrufen.

   Dieser Service konfiguriert Service Connect durch Hinzufügen des `serviceConnectConfiguration`-Parameters. Der Namespace ist nicht erforderlich, da für den Cluster ein Standard-Namespace konfiguriert ist. Client-Anwendungen, die in ECS im Namespace ausgeführt werden, stellen eine Verbindung zu diesem Service her, indem sie den `portName` und den Port im `clientAliases` verwenden. Dieser Service ist beispielsweise über `http://nginx:80/` erreichbar, da Nginx eine Willkommensseite im Stammverzeichnis `/` bereitstellt. Externe Anwendungen, die nicht in Amazon ECS ausgeführt werden oder sich nicht im selben Namespace befinden, können diese Anwendung über den Service-Connect-Proxy erreichen, indem sie die IP-Adresse der Aufgabe und die Portnummer aus der Aufgabendefinition verwenden. Fügen Sie für Ihre `tls` Konfiguration das Zertifikat `arn` für Ihre `awsPcaAuthorityArnkmsKey`, Ihre und `roleArn` Ihre IAM-Rolle hinzu.

   Dieser Service verwendet ein`logConfiguration`, um die Service-Connect-Proxyausgabe von `stdout` und `stderr` zu Amazon CloudWatch Logs zu senden. Diese Rolle zur Aufgabenausführung verfügt nicht über die zusätzlichen Berechtigungen, die für die Erstellung der Protokollgruppe CloudWatch Logs erforderlich sind. Erstellen Sie die Protokollgruppe in CloudWatch Logs mithilfe von AWS Management Console oder AWS CLI. Es wird empfohlen, diese Protokollgruppe zu erstellen und die CloudWatch Proxyprotokolle in Logs zu speichern. Wenn Sie die Proxyprotokolle nicht an Logs senden möchten, CloudWatch können Sie die entfernen`logConfiguration`.

```
{
    "cluster": "tutorial",
    "deploymentConfiguration": {
        "maximumPercent": 200,
```

```
        "minimumHealthyPercent": 0
    },
    "deploymentController": {
        "type": "ECS"
    },
    "desiredCount": 1,
    "enableECSManagedTags": true,
    "enableExecuteCommand": true,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [
                "sg-EXAMPLE"
            ],
            "subnets": [
                "subnet-EXAMPLE",
                "subnet-EXAMPLE",
                "subnet-EXAMPLE"
            ]
           }
    },
    "platformVersion": "LATEST",
    "propagateTags": "SERVICE",
    "serviceName": "service-connect-nginx-service",
    "serviceConnectConfiguration": {
        "enabled": true,
        "services": [
            {
                "portName": "nginx",
                "clientAliases": [
                    {
                        "port": 80
                    }
                ],
                "tls": {
                   "issuerCertificateAuthority": {
                      "awsPcaAuthorityArn": "certificateArn"
                   },
                   "kmsKey": "kmsKey",
                   "roleArn": "iamRoleArn"
                }
            }
        ],
```

```
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "service-connect-proxy"
            }
        }
    },
    "taskDefinition": "service-connect-nginx"
}
```

b. Erstellen Sie einen Dienst mit der `service-connect-nginx-service.json` Datei:

```
aws ecs create-service --cluster tutorial --cli-input-json file://service-
connect-nginx-service.json
```

Ausgabe:

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/tutorial/
service-connect-nginx-service",
        "serviceName": "service-connect-nginx-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
service-connect-nginx:1",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
                "rollback": false
            },
```

```
            "maximumPercent": 200,
            "minimumHealthyPercent": 0
        },
        "deployments": [
            {
                "id": "ecs-svc/3763308422771520962",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/service-connect-nginx:1",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": 1661210032.602,
                "updatedAt": 1661210032.602,
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "assignPublicIp": "ENABLED",
                        "securityGroups": [
                            "sg-EXAMPLE"
                        ],
                        "subnets": [
                            "subnet-EXAMPLEf",
                            "subnet-EXAMPLE",
                            "subnet-EXAMPLE"
                        ]
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-
svc/3763308422771520962 in progress.",
                "failedLaunchTaskCount": 0,
                "replacedTaskCount": 0,
                "serviceConnectConfiguration": {
                    "enabled": true,
                    "namespace": "service-connect",
                    "services": [
                        {
                            "portName": "nginx",
                            "clientAliases": [
                                {
```

```
                                    "port": 80
                                }
                            ]
                        }
                    ],
                    "logConfiguration": {
                        "logDriver": "awslogs",
                        "options": {
                            "awslogs-group": "/ecs/service-connect-proxy",
                            "awslogs-region": "us-west-2",
                            "awslogs-stream-prefix": "service-connect-proxy"
                        },
                        "secretOptions": []
                    }
                },
                "serviceConnectResources": [
                    {
                        "discoveryName": "nginx",
                        "discoveryArn": "arn:aws:servicediscovery:us-
west-2:123456789012:service/srv-EXAMPLE"
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "version": 0,
        "events": [],
        "createdAt": 1661210032.602,
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "assignPublicIp": "ENABLED",
                "securityGroups": [
                    "sg-EXAMPLE"
                ],
                "subnets": [
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE"
                ]
            }
        },
```

```
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": true,
        "propagateTags": "SERVICE",
        "enableExecuteCommand": true
    }
}
```

Die von Ihnen angegebene `serviceConnectConfiguration` wird in der ersten Bereitstellung der Ausgabe angezeigt. Wenn Sie Änderungen am ECS-Service vornehmen, die Änderungen an Aufgaben erfordern, wird von Amazon ECS eine neue Bereitstellung erstellt.

Schritt 3: Überprüfen der Möglichkeit, eine Verbindung herzustellen

Um zu überprüfen, ob Service Connect konfiguriert ist und funktioniert, führen Sie die folgenden Schritte aus, um von einer externen Anwendung aus eine Verbindung zum Webservice herzustellen. Sehen Sie sich dann die zusätzlichen Metriken an, CloudWatch die der Service Connect-Proxy erstellt.

So stellen Sie über eine externe Anwendung eine Verbindung mit dem Webservice her

- Herstellen einer Verbindung mit der IP-Adresse der Aufgabe und dem Container-Port mithilfe der IP-Adresse der Aufgabe

  Verwenden Sie die AWS CLI , um die Aufgaben-ID abzurufen, indem Sie den verwenden`aws ecs list-tasks --cluster tutorial`.

  Wenn Ihre Subnetze und Sicherheitsgruppe Datenverkehr aus dem öffentlichen Internet auf dem Port aus der Aufgabendefinition zulassen, können Sie von Ihrem Computer aus eine Verbindung mit der öffentlichen IP-Adresse herstellen. Die öffentliche IP ist jedoch nicht über `describe-tasks` verfügbar, sodass die Schritte beinhalten, zu Amazon EC2 zu gehen AWS Management Console oder die Details der AWS CLI elastic network interface.

  In diesem Beispiel verwendet eine Amazon-EC2-Instance in derselben VPC die private IP der Aufgabe. Die Anwendung ist Nginx, aber der `server: envoy`-Header zeigt, dass der Service-Connect-Proxy verwendet wird. Der Service-Connect-Proxy überwacht den Container-Port aus der Aufgabendefinition.

```
$ curl -v 10.0.19.50:80/
*   Trying 10.0.19.50:80...
* Connected to 10.0.19.50 (10.0.19.50) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.0.19.50
> User-Agent: curl/7.79.1
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< server: envoy
< date: Tue, 23 Aug 2022 03:53:06 GMT
< content-type: text/html
< content-length: 612
< last-modified: Tue, 16 Apr 2019 13:08:19 GMT
< etag: "5cb5d3c3-264"
< accept-ranges: bytes
< x-envoy-upstream-service-time: 0
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
```

```
</body>
</html>
```

So zeigen Sie Service-Connect-Metriken an

Der Service Connect-Proxy erstellt Anwendungsmetriken (HTTP-, HTTP2-, gRPC- oder TCP-Verbindung) in CloudWatch Metriken. Wenn Sie die CloudWatch Konsole verwenden, sehen Sie sich die zusätzlichen metrischen Dimensionen DiscoveryName, (DiscoveryName, ServiceName, ClusterName), TargetDiscoveryName und (TargetDiscoveryName, ServiceName, ClusterName) unter dem Amazon ECS-Namespace an. Weitere Informationen zu diesen Metriken und den Dimensionen finden Sie unter Verfügbare Metriken anzeigen im Amazon CloudWatch Logs-Benutzerhandbuch.

Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden

Ihr Amazon-ECS-Service kann optional so konfiguriert werden, dass er die Amazon-ECS-Serviceerkennung verwendet. Service Discovery verwendet AWS Cloud Map API-Aktionen, um HTTP- und DNS-Namespaces für Ihre Amazon ECS-Services zu verwalten. Weitere Informationen finden Sie unter Was ist AWS Cloud Map? im AWS Cloud Map -Entwicklerhandbuch.

Service Discovery ist in den folgenden Regionen verfügbar: AWS

Name der Region	Region
USA Ost (Nord-Virginia)	us-east-1
USA Ost (Ohio)	us-east-2
USA West (Nordkalifornien)	us-west-1
USA West (Oregon)	us-west-2
Afrika (Kapstadt)	af-south-1
Asien-Pazifik (Hongkong)	ap-east-1
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Hyderabad)	ap-south-2

Name der Region	Region
Asien-Pazifik (Tokio)	ap-northeast-1
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Osaka)	ap-northeast-3
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Asien-Pazifik (Jakarta)	ap-southeast-3
Asien-Pazifik (Melbourne)	ap-southeast-4
Kanada (Zentral)	ca-central-1
Kanada West (Calgary)	ca-west-1
China (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europa (Zürich)	eu-central-2
Europa (Irland)	eu-west-1
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3
Europa (Mailand)	eu-south-1
Europa (Stockholm)	eu-north-1
Israel (Tel Aviv)	il-central-1
Europa (Spanien)	eu-south-2

Name der Region	Region
Naher Osten (VAE)	me-central-1
Naher Osten (Bahrain)	me-south-1
Südamerika (São Paulo)	sa-east-1
AWS GovCloud (US-Ost)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

Konzepte für Service Discovery

Die Serviceerkennung umfasst folgende Komponenten:

- Service Discovery-Namespace: Eine logische Gruppe von Service Discovery-Services, die den gleichen Domain-Namen haben, zum Beispiel `example.com`. Dies ist der Domain-Name, an den Sie den Datenverkehr weiterleiten möchten. Sie können einen Namespace mit einem Aufruf des `aws servicediscovery create-private-dns-namespace` Befehls oder in der Amazon ECS-Konsole erstellen. Sie können den `aws servicediscovery list-namespaces`-Befehl zum Anzeigen der zusammenfassenden Informationen über die Namespaces, die vom aktuellen Konto erstellt wurden, verwenden. Weitere Informationen zu den Service Discovery-Befehlen finden Sie unter `create-private-dns-namespace` und `list-namespaces` im AWS CLI Referenzhandbuch AWS Cloud Map (Service Discovery).
- Service-Discovery Service: Ist im Service Discover-Namespace vorhanden und besteht aus dem Servicenamen und der DNS-Konfiguration für den Namespace. Er stellt die folgende Kernkomponente bereit:
  - Dienstregistrierung: Ermöglicht es Ihnen, über DNS- oder AWS Cloud Map API-Aktionen nach einem Dienst zu suchen und einen oder mehrere verfügbare Endpunkte wiederherzustellen, die für die Verbindung mit dem Dienst verwendet werden können.
- Service Discovery-Instance: Existiert innerhalb des Service Discovery-Service und besteht aus den Attributen, die jedem Amazon-ECS-Service im Serviceverzeichnis zugeordnet sind.
  - Instance-Attribute: Die folgenden Metadaten werden als benutzerdefinierte Attribute für jeden Amazon-ECS-Service hinzugefügt, der für die Verwendung von Service Discovery konfiguriert ist:

- **`AWS_INSTANCE_IPV4`**— Für einen `A` Datensatz die IPv4-Adresse, die Route 53 als Antwort auf DNS-Anfragen AWS Cloud Map zurückgibt und zurückgibt, wenn beispielsweise Instanzdetails entdeckt werden. `192.0.2.44`
- **`AWS_INSTANCE_PORT`**: Der Port-Wert, der dem Service Disovery-Service zugeordnet ist.
- **`AVAILABILITY_ZONE`**: Die Availability Zone, in der die Aufgabe gestartet wurde. Bei Aufgaben, die den Starttyp EC2 verwenden, ist dies die Availability Zone, in der die Container-Instance besteht. Bei Aufgaben, die den Starttyp Fargate verwenden, ist dies die Availability Zone, in der die Elastic-Network-Schnittstelle besteht.
- **`REGION`**: Die Region, in der sich die Aufgabe befindet.
- **`ECS_SERVICE_NAME`**: Der Name des Amazon-ECS-Services, zu dem die Aufgabe gehört.
- **`ECS_CLUSTER_NAME`**: Der Name des Amazon ECS-Clusters, zu dem die Aufgabe gehört.
- **`EC2_INSTANCE_ID`**: Die ID der Container-Instance, in der die Aufgabe platziert wurde. Dieses benutzerdefinierte Attribut wird nicht hinzugefügt, wenn die Aufgabe den Starttyp Fargate verwendet.
- **`ECS_TASK_DEFINITION_FAMILY`**: Die Aufgabendefinitionsfamilie, die die Aufgabe verwendet.
- **`ECS_TASK_SET_EXTERNAL_ID`**: Wenn eine Aufgabe für eine externe Bereitstellung erstellt und einer Service Discovery-Registrierung zugeordnet wird, dann enthält das Attribut `ECS_TASK_SET_EXTERNAL_ID` die externe ID des Aufgabensatzes.

- Amazon ECS-Zustandsprüfungen: Amazon ECS führt regelmäßige Zustandsprüfungen auf Container-Ebene durch. Wenn ein Endpunkt die Zustandsprüfung nicht besteht, wird er aus dem DNS-Routing entfernt und als fehlerhaft markiert.

Überlegungen zu Service Discovery

Bei der Verwendung der Serviceerkennung sollte Folgendes berücksichtigt werden:

- Service Discovery wird für Aufgaben auf Fargate unterstützt, die Plattform-Version v1.1.0 oder höher verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
- Services, die für die Verwendung von Service Discovery konfiguriert sind, haben ein Limit von 1000 Aufgaben pro Service. Dies ist auf eine Service-Quote der Route 53 zurückzuführen.

- Der Create Service-Workflow in der Amazon ECS-Konsole unterstützt nur die Registrierung von Services in private DNS-Namespaces. Wenn ein AWS Cloud Map privater DNS-Namespace erstellt wird, wird automatisch eine private gehostete Route 53-Zone erstellt.
- Die VPC DNS-Attribute müssen für eine erfolgreiche DNS-Auflösung konfiguriert werden. Weitere Informationen zum Konfigurieren der Attribute finden Sie unter DNS-Support für Ihre VPC im Amazon VPC-Benutzerhandbuch.
- Die für einen Service Discovery-Service erstellten DNS-Datensätze werden auch bei Verwendung von öffentlichen Namespaces immer mit der privaten IP-Adresse für die Aufgabe anstelle der öffentlichen IP-Adresse registriert.
- Service Discovery erfordert, dass Aufgaben entweder Netzwerkmodus `awsvpc`, `bridge` oder `host` angeben (`none` wird nicht unterstützt).
- Wenn die Service-Aufgabendefinition den `awsvpc`-Netzwerkmodus verwendet, können Sie für jede Service-Aufgabe eine beliebige Kombination aus `A`- oder SRV-Datensätzen erstellen. Wenn Sie SRV-Datensätze verwenden, ist ein Port erforderlich.
- Wenn die Service-Aufgabendefinition den Netzwerkmodus `bridge` oder `host` verwendet, wird nur der SRV-Datensatz als DNS-Datensatztyp unterstützt. Erstellen Sie einen SRV-Datensatz für jede Serviceaufgabe. Der SRV-Datensatz muss eine Kombination aus Containername und Container-Port aus der Aufgabendefinition angeben.
- DNS-Datensätze für einen Service zur Serviceerkennung können innerhalb Ihrer VPC abgefragt werden. Sie verwenden das folgende Format: `<service discovery service name>.<service discovery namespace>`.
- Wenn Sie eine DNS-Abfrage nach dem Namen des Services durchführen, geben `A`-Datensätze eine Reihe von IP-Adressen zurück, die Ihren Aufgaben entsprechen. SRV-Datensätze geben einen Satz von IP-Adressen und Ports für jede Aufgabe zurück.
- Wenn Sie über acht oder weniger fehlerfreie Datensätze verfügen, beantwortet Route 53 alle DNS-Abfragen mit allen fehlerfreien Datensätzen.
- Wenn alle Datensätze fehlerhaft sind, beantwortet Route 53 DNS-Abfragen mit bis zu acht fehlerhaften Datensätzen.
- Sie können die Serviceerkennung für einen Service konfigurieren, der sich hinter einem Load Balancer befindet, aber der Serviceerkennungsverkehr wird immer an die Aufgabe und nicht an den Load Balancer weitergeleitet.
- Service Discovery unterstützt die Verwendung von Classic Load Balancers nicht.
- Wir empfehlen Ihnen, Zustandsprüfungen auf Containerebene zu verwenden, die von Amazon ECS für Ihren Service zur Service Discovery verwaltet werden.

  - HealthCheckCustomConfig—Amazon ECS verwaltet Gesundheitschecks in Ihrem Namen. Amazon ECS verwendet Informationen aus Container- und Zustandsprüfungen sowie Ihren Aufgabenstatus, um den Zustand mit AWS Cloud Map zu aktualisieren. Dies wird beim Erstellen Ihres Service Discovery-Service mit dem Parameter `--health-check-custom-config` festgelegt. Weitere Informationen finden Sie HealthCheckCustomConfigin der AWS Cloud Map API-Referenz.
- Die AWS Cloud Map Ressourcen, die bei der Verwendung von Service Discovery erstellt werden, müssen manuell bereinigt werden.
- Aufgaben und Instanzen werden solange registriert, `UNHEALTHY` bis die Container-Integritätsprüfungen einen Wert zurückgeben. Wenn die Integritätsprüfungen erfolgreich sind, wird der Status auf aktualisiert`HEALTHY`. Wenn die Container-Integritätsprüfungen fehlschlagen, wird die Registrierung der Service Discovery-Instance aufgehoben.

Service Discovery-Preisgestaltung

Den Kunden, die Amazon-ECS-Service Discovery nutzen, werden die Route 53-Ressourcen und die AWS Cloud Map -Discovery API-Operationen berechnet. Dies umfasst die Kosten für das Erstellen der von Route 53 gehosteten Zonen und Abfragen der Service-Registry. Weitere Informationen finden Sie unter AWS Cloud Map -Preisgestaltung im AWS Cloud Map -Entwicklerhandbuch.

Amazon ECS führt Integritätsprüfungen auf Containerebene durch und macht sie für AWS Cloud Map benutzerdefinierte API-Operationen zur Integritätsprüfung verfügbar. Dies wird den Kunden derzeit ohne Mehrkosten zur Verfügung gestellt. Wenn Sie zusätzliche Zustandsprüfungen für öffentlich zugängliche Aufgaben konfigurieren, werden Ihnen diese Zustandsprüfungen in Rechnung gestellt.

Einen Amazon ECS-Service erstellen, der Service Discovery verwendet

Erfahren Sie, wie Sie einen Service erstellen, der eine Fargate-Aufgabe enthält, die Service Discovery mit dem AWS CLI verwendet.

Eine Liste AWS-Regionen dieser Support Service Discovery finden Sie unterVerwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.

Weitere Informationen über Regionen, die Fargate unterstützen, finden Sie unter the section called “AWS Fargate-Regionen”.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.
- Die unter Einrichtung für die Verwendung von Amazon ECS beschriebenen Schritte sind abgeschlossen.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben mindestens eine VPC und eine Sicherheitsgruppe erstellt. Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.

Schritt 1: Erstellen Sie die Service Discovery-Ressourcen in AWS Cloud Map

Führen Sie die folgenden Schritte aus, um Ihren Service-Discovery-Namespace und Service zur Service Discovery zu erstellen:

1. Erstellen Sie einen privaten Namespace für die Service Discovery der Cloud Map. In diesem Beispiel wird ein Namespace mit dem Namen `tutorial` erstellt. Ersetzen Sie *vpc-abcd1234* mit der ID eines Ihrer vorhandenen VPCs.

   ```
   aws servicediscovery create-private-dns-namespace \
         --name tutorial \
         --vpc vpc-abcd1234
   ```

   Die Ausgabe dieses Befehls sieht wie folgt aus.

   ```
   {
       "OperationId": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e"
   }
   ```

2. Überprüfen Sie mithilfe der `OperationId` aus der Ausgabe des vorherigen Schritts, ob der private Namespace erfolgreich erstellt wurde. Notieren Sie sich die Namespace-ID, da Sie sie in nachfolgenden Befehlen verwenden.

   ```
   aws servicediscovery get-operation \
   ```

```
      --operation-id h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e
```

Die Ausgabe sieht wie folgt aus.

```
{
    "Operation": {
        "Id": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1519777852.502,
        "UpdateDate": 1519777856.086,
        "Targets": {
           "NAMESPACE": "ns-uejictsjen2i4eeg"
        }
    }
}
```

3. Erstellen Sie mithilfe der NAMESPACE-ID aus der Ausgabe des vorherigen Schritts einen Service zur Service Discovery. In diesem Beispiel wird ein Service mit dem Namen `myapplication` erstellt. Notieren Sie sich die Service-ID und den ARN, da Sie sie in nachfolgenden Befehlen verwenden.

```
aws servicediscovery create-service \
      --name myapplication \
      --dns-config "NamespaceId="ns-
uejictsjen2i4eeg",DnsRecords=[{Type="A",TTL="300"}]" \
      --health-check-custom-config FailureThreshold=1
```

Die Ausgabe sieht wie folgt aus.

```
{
    "Service": {
       "Id": "srv-utcrh6wavdkggqtk",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:service/srv-
utcrh6wavdkggqtk",
        "Name": "myapplication",
        "DnsConfig": {
            "NamespaceId": "ns-uejictsjen2i4eeg",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
```

```
                }
            ]
        },
        "HealthCheckCustomConfig": {
            "FailureThreshold": 1
        },
        "CreatorRequestId": "e49a8797-b735-481b-a657-b74d1d6734eb"
    }
}
```

Schritt 2: Erstellen der Amazon-ECS-Ressourcen

Führen Sie die folgenden Schritte aus, um Ihren Amazon-ECS-Cluster, die Definition der Aufgabe und den Service zu erstellen:

1. Erstellen Sie einen Amazon-ECS-Cluster. In diesem Beispiel wird ein Cluster mit dem Namen `tutorial` erstellt.

```
aws ecs create-cluster \
      --cluster-name tutorial
```

2. Registrieren Sie eine Aufgabendefinition, die mit Fargate kompatibel ist und den `awsvpc`-Netzwerkmodus verwendet. Dazu gehen Sie wie folgt vor:

   a. Erstellen Sie eine Datei mit dem Namen `fargate-task.json` mit dem Inhalt der folgenden Aufgabendefinition.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "httpd:2.4",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
```

```
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample
 App</h1> <h2>Congratulations!</h2> <p>Your application is now running on a
 container in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/
htdocs/index.html && httpd-foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

b. Registrieren Sie die Aufgabendefinition mithilfe von `fargate-task.json`.

```
aws ecs register-task-definition \
      --cli-input-json file://fargate-task.json
```

3. Erstellen Sie einen ECS Service, indem Sie die folgenden Schritte ausführen:

a. Erstellen Sie eine Datei mit dem Namen `ecs-service-discovery.json`, die den Inhalt des ECS-Services enthält, den Sie erstellen wollen. Dieses Beispiel verwendet die Aufgabendefinition, die im vorherigen Schritt erstellt wurde. Ein `awsvpcConfiguration` ist erforderlich, da die Beispiel-Aufgabendefinition den `awsvpc`-Netzwerkmodus verwendet.

Wenn Sie den ECS-Service erstellen, geben Sie den Fargate-Starttyp und die `LATEST`-Plattformversion an, die Service Discovery unterstützt. Wenn der Service zur Service Discovery in AWS Cloud Map erstellt wird, ist `registryArn` der zurückgegebene ARN. Die `securityGroups` und `subnets` muss zu der VPC gehören, die zum Erstellen des Cloud Map-Namespaces verwendet wird. Sie können die Sicherheitsgruppen- und Subnetz-IDs über die VPC-Konsole abrufen.

```
{
```

```
    "cluster": "tutorial",
    "serviceName": "ecs-service-discovery",
    "taskDefinition": "tutorial-task-def",
    "serviceRegistries": [
       {
          "registryArn":
 "arn:aws:servicediscovery:region:aws_account_id:service/srv-utcrh6wavdkggqtk"
       }
    ],
    "launchType": "FARGATE",
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234" ]
       }
    },
    "desiredCount": 1
}
```

b. Erstellen Sie Ihren ECS-Service mithilfe von `ecs-service-discovery.json`.

```
aws ecs create-service \
      --cli-input-json file://ecs-service-discovery.json
```

Schritt 3: Überprüfen Sie Service Discovery in AWS Cloud Map

Sie können überprüfen, ob alles ordnungsgemäß erstellt wurde, indem Sie Ihre Service-Discovery-Informationen abfragen. Nachdem die Diensterkennung konfiguriert wurde, können Sie entweder AWS Cloud Map API-Operationen verwenden oder `dig` von einer Instanz in Ihrer VPC aus aufrufen. Dazu gehen Sie wie folgt vor:

1. Listen Sie mithilfe der Service-Discovery-Service-ID die Service-Discovery-Instances auf. Notieren Sie sich die Instance-ID (fett markiert) für die Ressourcen-Bereinigung.

```
aws servicediscovery list-instances \
      --service-id srv-utcrh6wavdkggqtk
```

Die Ausgabe sieht wie folgt aus.

```
{
    "Instances": [
        {
            "Id": "16becc26-8558-4af1-9fbd-f81be062a266",
            "Attributes": {
                "AWS_INSTANCE_IPV4": "172.31.87.2"
                "AWS_INSTANCE_PORT": "80",
                "AVAILABILITY_ZONE": "us-east-1a",
                "REGION": "us-east-1",
                "ECS_SERVICE_NAME": "ecs-service-discovery",
                "ECS_CLUSTER_NAME": "tutorial",
                "ECS_TASK_DEFINITION_FAMILY": "tutorial-task-def"
            }
        }
    ]
}
```

2. Verwenden Sie den Service-Discovery-Namespace, den Service und zusätzliche Parameter wie den ECS-Clusternamen, um Details zu den Service-Discovery-Instances abzufragen.

```
aws servicediscovery discover-instances \
      --namespace-name tutorial \
      --service-name myapplication \
      --query-parameters ECS_CLUSTER_NAME=tutorial
```

3. Die DNS-Einträge, die in der von Route 53 gehosteten Zone für den Service zur Service Discovery erstellt wurden, können mit den folgenden AWS CLI -Befehlen abgefragt werden:

   a. Rufen Sie mithilfe der Namespace-ID Informationen zum Namespace ab, die die von Route 53 gehostete Zonen-ID enthalten.

   ```
   aws servicediscovery \
         get-namespace --id ns-uejictsjen2i4eeg
   ```

   Die Ausgabe sieht wie folgt aus.

   ```
   {
       "Namespace": {
           "Id": "ns-uejictsjen2i4eeg",
           "Arn": "arn:aws:servicediscovery:region:aws_account_id:namespace/ns-
   uejictsjen2i4eeg",
   ```

```
        "Name": "tutorial",
        "Type": "DNS_PRIVATE",
        "Properties": {
             "DnsProperties": {
                "HostedZoneId": "Z35JQ4ZFDRYPLV"
            }
        },
        "CreateDate": 1519777852.502,
        "CreatorRequestId": "9049a1d5-25e4-4115-8625-96dbda9a6093"
    }
}
```

b. Verwenden Sie die ID der gehosteten Route-53-Zone aus dem vorherigen Schritt (siehe fett gedruckten Text), um den Ressourcendatensatz für die gehostete Zone abzurufen.

```
aws route53 list-resource-record-sets \
      --hosted-zone-id Z35JQ4ZFDRYPLV
```

4. Sie können den DNS auch von einer Instance innerhalb Ihrer VPC mit `dig` abfragen.

```
dig +short myapplication.tutorial
```

Schritt 4: Bereinigen

Wenn Sie mit diesem Tutorial fertig sind, bereinigen Sie die zugeordneten Ressourcen, um Gebühren für ungenutzte Ressourcen zu vermeiden. Dazu gehen Sie wie folgt vor:

1. Deregistrieren Sie die Service Discovery Service-Instances mithilfe der zuvor notierten Service-ID und Instance-ID.

```
aws servicediscovery deregister-instance \
      --service-id srv-utcrh6wavdkggqtk \
      --instance-id 16becc26-8558-4af1-9fbd-f81be062a266
```

Die Ausgabe sieht wie folgt aus.

```
{
    "OperationId": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv"
}
```

2. Überprüfen Sie mithilfe von `OperationId` aus der Ausgabe des vorherigen Schritts, ob die Service Discovery-Instances erfolgreich deregistriert wurden.

```
aws servicediscovery get-operation \
      --operation-id xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv
```

```
{
  "Operation": {
        "Id": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv",
        "Type": "DEREGISTER_INSTANCE",
        "Status": "SUCCESS",
        "CreateDate": 1525984073.707,
        "UpdateDate": 1525984076.426,
        "Targets": {
            "INSTANCE": "16becc26-8558-4af1-9fbd-f81be062a266",
            "ROUTE_53_CHANGE_ID": "C5NSRG1J4I1FH",
            "SERVICE": "srv-utcrh6wavdkggqtk"
        }
    }
}
```

3. Löschen Sie den Service zur Service Discovery unter Verwendung der Service-ID.

```
aws servicediscovery delete-service \
      --id srv-utcrh6wavdkggqtk
```

4. Löschen Sie den Service Discovery-Namespace mithilfe der Namespace-ID.

```
aws servicediscovery delete-namespace \
      --id ns-uejictsjen2i4eeg
```

Die Ausgabe sieht wie folgt aus.

```
{
    "OperationId": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj"
}
```

5. Überprüfen Sie mithilfe der `OperationId` aus der vorherigen Ausgabe des vorherigen Schritts, ob der Service-Discovery-Namespace erfolgreich gelöscht wurde.

```
aws servicediscovery get-operation \
```

```
      --operation-id c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj
```

Die Ausgabe sieht wie folgt aus.

```
{
    "Operation": {
        "Id": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj",
        "Type": "DELETE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1525984602.211,
        "UpdateDate": 1525984602.558,
        "Targets": {
            "NAMESPACE": "ns-rymlehshst7hhukh",
            "ROUTE_53_CHANGE_ID": "CJP2A2M86XW3O"
        }
    }
}
```

6. Aktualisieren Sie die gewünschte Anzahl für den Amazon-ECS-Service auf `0`. Sie müssen dies tun, um den Service im nächsten Schritt zu löschen.

```
aws ecs update-service \
      --cluster tutorial \
      --service ecs-service-discovery \
      --desired-count 0
```

7. Löschen Sie den Amazon-ECS-Service.

```
aws ecs delete-service \
      --cluster tutorial \
      --service ecs-service-discovery
```

8. Löschen Sie den Amazon-ECS-Cluster.

```
aws ecs delete-cluster \
      --cluster tutorial
```

Schützen Sie Ihre Amazon ECS-Aufgaben davor, durch Scale-In-Ereignisse beendet zu werden

Sie können den Task-Scale-In-Schutz von Amazon ECS verwenden, um zu verhindern, dass Ihre Aufgaben durch Scale-In-Ereignisse von Service Auto Scaling oder Bereitstellungen beendet werden.

Bestimmte Anwendungen erfordern einen Mechanismus zum Schutz unternehmenskritischer Aufgaben vor der Beendigung durch Abskalierungsereignisse in Zeiten geringer Auslastung oder während Service-Bereitstellungen. Beispielsweise:

- Sie verfügen über eine asynchrone Anwendung mit Warteschlangenverarbeitung, z. B. einen Video-Transkodierungsauftrag, bei dem einige Aufgaben stundenlang ausgeführt werden müssen, selbst wenn die kumulierte Service-Auslastung gering ist.
- Sie haben eine Spieleanwendung, die Spieleserver als Amazon ECS-Aufgaben ausführt, die auch dann weiterlaufen müssen, wenn sich alle Benutzer abgemeldet haben, um die Startlatenz bei einem Serverneustart zu reduzieren.
- Wenn Sie eine neue Codeversion bereitstellen, müssen Aufgaben weiterhin ausgeführt werden, da eine erneute Verarbeitung kostenintesiv wäre.

Um zu verhindern, dass Aufgaben, die zu Ihrem Service gehören, bei einem Abskalierungs-Ereignis beendet werden, setzen Sie das `protectionEnabled`-Attribut auf `true`. Standardmäßig sind Aufgaben für 2 Stunden geschützt. Sie können den Schutzzeitraum mithilfe des `expiresInMinutes`-Attributs anpassen. Sie können Ihre Aufgaben für mindestens 1 Minute und bis zu maximal 2 880 Minuten (48 Stunden) schützen.

Nachdem eine Aufgabe ihre erforderliche Arbeit beendet hat, können Sie das `protectionEnabled`-Attribut auf `false` setzen, sodass die Aufgabe durch nachfolgende Abskalierungsereignisse beendet werden kann.

Mechanismus des Abskalierungsschutzes für Aufgaben

Sie können den Abskalierungsschutz für Aufgaben entweder über den Amazon-ECS-Container-Agent-Endpunkt oder die Amazon-ECS-API einrichten und abrufen.

- Amazon-ECS-Container-Agent-Endpunkt

Wir empfehlen die Verwendung des Amazon-ECS-Container-Agent-Endpunkts für Aufgaben, die den Schutzbedarf selbst bestimmen können. Verwenden Sie diesen Ansatz für warteschlangenbasierte oder Auftragsverarbeitungs-Workloads.

Wenn ein Container mit der Verarbeitung von Aufgaben beginnt, z. B. durch Konsumieren einer SQS-Nachricht, können Sie das `ProtectionEnabled`-Attribut über den Endpunkt-Pfad `$ECS_AGENT_URI/task-protection/v1/state` zum Abskalierungsschutz für Aufgaben innerhalb des Containers festlegen. Amazon ECS beendet diese Aufgabe bei Abskalierungsereignissen nicht. Nachdem Ihre Aufgabe ihre Arbeit beendet hat, können Sie das `ProtectionEnabled` Attribut mit demselben Endpunkt löschen, sodass die Aufgabe bei nachfolgenden Scale-In-Ereignissen beendet werden kann.

Weitere Informationen zum Amazon ECS-Container-Agent-Endpunkt finden Sie unterSkalierbarer Schutzendpunkt für Amazon ECS Task.

- Amazon-ECS-API

  Sie können die Amazon ECS-API verwenden, um den Task-Scale-In-Schutz einzurichten und abzurufen, wenn Ihre Anwendung über eine Komponente verfügt, die den Status aktiver Aufgaben verfolgt. Verwenden Sie `UpdateTaskProtection`, um eine oder mehrere Aufgaben als geschützt zu markieren. Wird verwendet`GetTaskProtection`, um den Schutzstatus abzurufen.

  Ein Beispiel für diesen Ansatz wäre, wenn Ihre Anwendung Spielserver-Sitzungen als Amazon-ECS-Aufgaben hostet. Wenn sich ein Benutzer bei einer Sitzung auf dem Server (Aufgabe) anmeldet, können Sie die Aufgabe als geschützt markieren. Nachdem sich der Benutzer abgemeldet hat, können Sie entweder den Schutz speziell für diese Aufgabe aufheben oder den Schutz für ähnliche Aufgaben, die keine aktiven Sitzungen mehr haben, regelmäßig aufheben, je nachdem, ob Sie Server im Leerlauf halten möchten.

  Weitere Informationen finden Sie unter UpdateTaskSchutz und GetTaskSchutz in der Amazon Elastic Container Service API-Referenz.

Sie können beide Ansätze kombinieren. Verwenden Sie beispielsweise den Amazon-ECS-Agent-Endpunkt, um den Aufgabenschutz innerhalb eines Containers einzurichten, und verwenden Sie die Amazon-ECS-API, um den Aufgabenschutz für Ihren externen Controller-Service zu entfernen.

Überlegungen

Berücksichtigen Sie die folgenden Punkte, bevor Sie den Abskalierungsschutz für Aufgaben verwenden:

- Wir empfehlen die Verwendung des Amazon-ECS-Container-Agent-Endpunkts, da der Amazon-ECS-Agent über integrierte Wiederholungsmechanismen und eine einfachere Schnittstelle verfügt.
- Sie können den Ablaufzeitraum für den Abskalierungsschutz für Aufgaben zurücksetzen, indem Sie `UpdateTaskProtection` für eine Aufgabe aufrufen, für die der Schutz bereits aktiviert ist.
- Bestimmen Sie, wie lange eine Aufgabe benötigen würde, um ihre erforderliche Arbeit abzuschließen, und legen Sie die `expiresInMinutes`-Eigenschaft entsprechend fest. Wenn Sie den Ablauf des Schutzes länger als nötig festlegen, entstehen Ihnen Kosten und Verzögerungen bei der Bereitstellung neuer Aufgaben.
- Der Aufgaben-Abskalierungsschutz wird auf dem Amazon-ECS-Container-Agenten `1.65.0` oder höher unterstützt.

  Sie können Unterstützung für dieses Feature auf Amazon EC2-Instances hinzufügen, indem Sie ältere Versionen der Amazon-ECS-Container-Agenten auf die aktuelle Version aktualisieren. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.
- Überlegungen zur Bereitstellung:
  - Wenn der Service eine fortlaufende Aktualisierung verwendet, werden neue Aufgaben erstellt, aber Aufgaben mit älteren Versionen werden nicht beendet, bis `protectionEnabled` gelöscht wird oder abläuft. Sie können den `maximumPercentage`-Parameter in der Bereitstellungskonfiguration auf einen Wert anpassen, der es ermöglicht, neue Aufgaben zu erstellen, wenn alte Aufgaben geschützt sind.
  - Wenn eine Blau/Grün-Aktualisierung angewendet wird, wird die blaue Bereitstellung mit geschützten Aufgaben nicht entfernt, wenn Aufgaben über `protectionEnabled` verfügen. Der Traffic wird auf die neuen Aufgaben umgeleitet, die auftauchen, und ältere Aufgaben werden erst entfernt, wenn sie gelöscht wurden oder abgelaufen `protectionEnabled` sind. Je nach Timeout der CodeDeploy CloudFormation OR-Updates kann es bei der Bereitstellung zu einem Timeout kommen und die älteren Blue-Aufgaben sind möglicherweise noch vorhanden.
  - Wenn Sie dies verwenden CloudFormation, hat der Update-Stack ein Timeout von 3 Stunden. Wenn Sie also Ihren Task-Schutz auf mehr als 3 Stunden einstellen, kann Ihre CloudFormation Implementierung zu einem Ausfall und Rollback führen.

Während der Zeit, in der Ihre alten Aufgaben geschützt sind, wird der CloudFormation Stack angezeigt`UPDATE_IN_PROGRESS`. Wenn der Abskalierungsschutz für Aufgaben entfernt wird oder innerhalb des 3-Stunden-Fensters abläuft, wird Ihre Bereitstellung erfolgreich sein und in den Status `UPDATE_COMPLETE` wechseln. Wenn die Bereitstellung länger als 3 Stunden in `UPDATE_IN_PROGRESS` verharrt, schlägt sie fehl, zeigt den `UPDATE_FAILED`-Status an und wird dann auf den alten Aufgabensatz zurückgesetzt.

- Amazon ECS sendet Service-Ereignisse, wenn geschützte Aufgaben eine Bereitstellung (fortlaufend oder blau/grün) davon abhalten, den stabilen Zustand zu erreichen, so dass Sie Abhilfemaßnahmen ergreifen können. Wenn Sie beim Versuch, den Schutzstatus einer Aufgabe zu aktualisieren, eine `DEPLOYMENT_BLOCKED`-Fehlermeldung erhalten, bedeutet dies, dass der Service über mehr geschützte Aufgaben verfügt als die gewünschte Anzahl von Aufgaben für den Service. Führen Sie einen der folgenden Schritte aus, um diesen Fehler zu beheben:
  - Warten Sie, bis der aktuelle Aufgabenschutz abgelaufen ist. Stellen Sie dann den Aufgabenschutz ein.
  - Stellen Sie fest, welche Aufgaben angehalten werden können. Dann verwenden Sie `UpdateTaskProtection` mit der auf `false` festgelegten `protectionEnabled`-Option für diese Aufgaben.
  - Erhöhen Sie die Anzahl der gewünschten Aufgaben des Services auf mehr als die Anzahl der geschützten Aufgaben.

Für den Abskalierungsschutz für Aufgaben erforderliche IAM-Berechtigungen

Die Aufgabe muss über die Amazon ECS-Aufgabenrolle mit den folgenden Berechtigungen verfügen:

- `ecs:GetTaskProtection`: Erlaubt dem Amazon-ECS-Container-Agenten `GetTaskProtection` aufzurufen.
- `ecs:UpdateTaskProtection`: Erlaubt dem Amazon-ECS-Container-Agenten `UpdateTaskProtection` aufzurufen.

Skalierbarer Schutzendpunkt für Amazon ECS Task

Der Amazon-ECS-Container-Agent fügt die `ECS_AGENT_URI`-Umgebungsvariable automatisch in die Amazon-ECS-Aufgabencontainer ein, um eine Methode für die Interaktion mit dem Container-Agent-API-Endpunkt bereitzustellen.

Wir empfehlen die Verwendung des Amazon-ECS-Container-Agent-Endpunkts für Aufgaben, die den Schutzbedarf selbst bestimmen können.

Wenn ein Container mit der Verarbeitung von Aufgaben beginnt, können Sie das `protectionEnabled` Attribut mithilfe des Pfads `$ECS_AGENT_URI/task-protection/v1/state` des Task-Scale-in-Schutz-Endpunkts innerhalb des Containers festlegen.

Verwenden Sie innerhalb eines Containers eine PUT-Anforderung an diesen URI, um den Task-Scale-In-Schutz einzurichten. Eine GET-Anfrage an diesen URI gibt den aktuellen Schutzstatus einer Aufgabe zurück.

Scale-in-Schutzanforderungsparameter für Aufgaben

Sie können den Abskalierungsschutz für Aufgaben mithilfe des `${ECS_AGENT_URI}/task-protection/v1/state`-Endpunkts mit den folgenden Anfragenparametern einrichten.

`ProtectionEnabled`

Geben Sie `true` an, ob eine Aufgabe als geschützt markiert werden soll. Geben Sie `false` an, dass der Schutz aufgehoben und die Aufgabe beendet werden kann.

Typ: Boolesch

Erforderlich: Ja

`ExpiresInMinutes`

Die Anzahl der Minuten, für die die Aufgabe geschützt ist. Sie können mindestens 1 Minute bis zu 2 880 Minuten (48 Stunden) angeben. Während dieses Zeitraums wird Ihre Aufgabe nicht durch Abskalierungsereignisse des Services Auto Scaling oder durch Bereitstellungen beendet. Nach Ablauf dieses Zeitraums wird der `protectionEnabled`-Parameter auf `false` gesetzt.

Wenn Sie keinen Zeitraum angeben, wird die Aufgabe automatisch für 120 Minuten (2 Stunden) geschützt.

Typ: Ganzzahl

Erforderlich: Nein

Die folgenden Beispiele zeigen, wie Sie den Aufgabenschutz mit unterschiedlichen Laufzeiten festlegen können.

Beispiel für den Schutz einer Aufgabe mit dem Standardzeitraum

Dieses Beispiel zeigt, wie eine Aufgabe mit dem Standardzeitraum von 2 Stunden geschützt wird.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Beispiel für den Schutz einer Aufgabe für 60 Minuten

Dieses Beispiel zeigt, wie eine Aufgabe mit dem `expiresInMinutes`-Parameter 60 Minuten lang geschützt wird.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Beispiel dafür, wie eine Aufgabe 24 Stunden lang geschützt werden kann

Dieses Beispiel zeigt, wie eine Aufgabe mit dem `expiresInMinutes`-Parameter 24 Stunden lang geschützt wird.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

Die PUT-Anforderung gibt die folgende Antwort zurück.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
}
```

Antwortparameter für den Task-Scale-In-Schutz

Die folgenden Informationen werden in der JSON-Antwort des Endpunkts zum Abskalierungsschutz für Aufgaben `${ECS_AGENT_URI}/task-protection/v1/state` zurückgegeben.

`ExpirationDate`

Die Epochenzeit, zu der der Schutz für die Aufgabe abläuft. Wenn die Aufgabe nicht geschützt ist, ist dieser Wert Null.

`ProtectionEnabled`

Der Schutzstatus der Aufgabe. Wenn der Abskalierungsschutz für eine Aufgabe aktiviert ist, ist der Wert `true`. Andernfalls ist es `false`.

`TaskArn`

Der vollständige Amazon Resource Name (ARN) der Aufgabe, zu der der Container gehört.

Das folgende Beispiel zeigt die für eine geschützte Aufgabe zurückgegebenen Details.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

Die folgenden Informationen werden zurückgegeben, wenn ein Fehler auftritt.

`Arn`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe.

`Detail`

Die auf den Fehler bezogenen Details.

`Reason`

Der Grund für den Fehlschlag.

Das folgende Beispiel zeigt die für eine nicht geschützte Aufgabe zurückgegebenen Details.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
        "Detail":null,
```

```
        "Reason":"TASK_NOT_VALID"
    }
}
```

Die folgenden Informationen werden zurückgegeben, wenn eine Ausnahme auftritt.

`requestID`

Die AWS Anforderungs-ID für den Amazon ECS-API-Aufruf, der zu einer Ausnahme führt.

`Arn`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe oder des Services.

`Code`

Der Fehlercode.

`Message`

Die Fehlermeldung.

> **Note**
>
> Wenn ein `RequestError`- oder `RequestTimeout`-Fehler auftritt, ist es wahrscheinlich, dass es sich um ein Netzwerkproblem handelt. Versuchen Sie, VPC-Endpunkte für Amazon ECS zu verwenden.

Das folgende Beispiel zeigt die Details, die zurückgegeben werden, wenn ein Fehler auftritt.

```
{
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

Der folgende Fehler wird angezeigt, wenn der Amazon-ECS-Agent aus Gründen wie Netzwerkproblemen oder einem Ausfall der Amazon-ECS-Steuerebene keine Antwort vom Amazon-ECS-Endpunkt erhalten kann.

```
{
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

Der folgende Fehler wird angezeigt, wenn der Amazon-ECS-Agent eine Drosselungsausnahme von Amazon ECS erhält.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Drosselungslogik für Amazon ECS-Services

Der Amazon-ECS-Service-Scheduler enthält eine Logik, die eine Begrenzung einführt, wie oft Serviceaufgaben gestartet werden, wenn sie wiederholt nicht gestartet werden können.

Wenn Aufgaben für einen Dienst wiederholt nicht in den `RUNNING` Status wechseln (also direkt von einem in einen `PENDING` `STOPPED` Status wechseln), wird die Zeit zwischen aufeinanderfolgenden Neustartversuchen schrittweise auf maximal 27 Minuten erhöht. Diese Höchstdauer kann sich in Zukunft ändern. Durch dieses Verhalten werden die Auswirkungen von fehlschlagenden Aufgaben auf Ihre Amazon-ECS-Cluster-Ressourcen oder die Fargate-Infrastrukturkosten reduziert. Wenn Ihr Service die Drosselungslogik initiiert, erhalten Sie die folgende Service-Ereignismeldung:

```
(service service-name) is unable to consistently start tasks successfully.
```

Amazon ECS hindert einen fehlgeschlagenen Service niemals daran, es erneut zu versuchen. Es wird auch nicht versucht, es auf andere Weise zu verändern, außer der Erhöhung der Zeit zwischen

den Neustarts. Die Service-Drosselungslogik stellt keine vom Benutzer einstellbaren Parameter bereit.

Wenn Sie Ihren Service auf eine neue Aufgabendefinition aktualisieren, kehrt der Service sofort zu einem normalen, ungedrosselten Zustand zurück. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.

Im Folgenden sind einige der häufigsten Ursachen aufgeführt, die diese Logik auslösen. Wir empfehlen, dass Sie manuelle Maßnahmen ergreifen, um das Problem zu beheben:

- Fehlende Ressourcen für das Hosten Ihrer Aufgabe, z. B. Ports, Speicher oder CPU-Einheiten in Ihrem Cluster. In diesem Fall wird auch eine Benachrichtigung über unzureichende Ressourcen angezeigt.
- Der Amazon-ECS-Container-Agent kann Ihr Docker-Image für die Aufgabe nicht abrufen. Dies kann an einem ungültigen Container-Image-Namen, -Image oder -Tag oder an einem Mangel an privater Registry-Authentifizierung oder -Berechtigungen liegen. In diesem Fall wird auch `CannotPullContainerError` in Ihren Fehlern für gestoppte Aufgaben angezeigt.
- Ungenügender Speicherplatz in Ihrer Container-Instance zum Erstellen des Containers. In diesem Fall wird auch `CannotCreateContainerError` in Ihren Fehlern für gestoppte Aufgaben angezeigt. Weitere Informationen finden Sie unter Fehlerbehebung beim Docker `API error (500): devmapper` in Amazon ECS.

> **Important**
>
> Aufgaben, die angehalten werden, nachdem sie den `RUNNING`-Status erreicht haben, starten weder die Drosselungslogik noch die zugehörige Service-Ereignismeldung. Wenn beispielsweise die Zustandsprüfungen von Elastic Load Balancing für einen Service fehlgeschlagen sind, wird eine Aufgabe als fehlerhaft eingestuft und Amazon ECS deregistriert sie und stoppt die Aufgabe. Zu diesem Zeitpunkt werden die Aufgaben nicht gedrosselt. Auch wenn ein Container-Befehl einer Aufgabe sofort mit einem Beendigungscode ungleich Null beendet wird, erhält die Aufgabe bereits den `RUNNING`-Status. Aufgaben, die sofort fehlschlagen, weil Befehlsfehler nicht die Drosselung oder die Serviceereignismeldung verursachen.

Parameter der Amazon ECS-Servicedefinition

Eine Servicedefinition legt fest, wie der Amazon-ECS-Service ausgeführt werden soll. Die folgenden Parameter können in einer Servicedefinition angegeben werden.

Starttyp

`launchType`

Typ: Zeichenfolge

Zulässige Werte: `EC2` | `FARGATE` | `EXTERNAL`

Erforderlich: Nein

Der Starttyp, der für die Ausführung Ihres Service verwendet wird. Ist kein Starttyp angegeben, wird standardmäßig `capacityProviderStrategy` verwendet. Weitere Informationen finden Sie unter Amazon-ECS-Starttypen.

Wenn eine `launchType` angegeben ist, muss der `capacityProviderStrategy`-Parameter weggelassen werden.

Kapazitätsanbieterstrategie

`capacityProviderStrategy`

Typ: Array von -Objekten

Erforderlich: Nein

Die Kapazitätsanbieterstrategie, die für den Service verwendet werden soll.

Eine Kapazitätsanbieterstrategie besteht aus einem oder mehreren Kapazitätsanbietern sowie dem ihnen zuzuweisenden `base`- und `weight`-Wert. Ein Kapazitätsanbieter muss dem Cluster zugeordnet sein, um in einer Kapazitätsanbieterstrategie verwendet werden zu können. Die PutClusterCapacityProviders API wird verwendet, um einen Kapazitätsanbieter mit einem Cluster zu verknüpfen. Es können nur Kapazitätsanbieter mit dem Status `ACTIVE` oder `UPDATING` verwendet werden.

Wenn eine `capacityProviderStrategy` angegeben ist, muss der `launchType`-Parameter weggelassen werden. Wenn keine `capacityProviderStrategy` oder kein `launchType` angegeben ist, wird die `defaultCapacityProviderStrategy` für den Cluster verwendet.

Wenn Sie einen Kapazitätsanbieter angeben wollen, der eine Auto-Scaling-Gruppe verwendet, muss der Kapazitätsanbieter bereits erstellt sein. Neue Kapazitätsanbieter können mit dem CreateCapacityProvider API-Vorgang erstellt werden.

Um einen AWS Fargate-Kapazitätsanbieter zu verwenden, geben Sie entweder den `FARGATE` oder den `FARGATE_SPOT` Kapazitätsanbieter an. Die AWS Fargate-Kapazitätsanbieter sind für alle Konten verfügbar und müssen nur mit einem Cluster verknüpft werden, um verwendet zu werden.

Die PutClusterCapacityProviders API-Operation wird verwendet, um die Liste der verfügbaren Kapazitätsanbieter für einen Cluster zu aktualisieren, nachdem der Cluster erstellt wurde.

`capacityProvider`

Typ: Zeichenfolge

Erforderlich: Ja

Der Kurzname oder vollständige Amazon-Ressourcenname (ARN) des Kapazitätsanbieters.

`weight`

Typ: Ganzzahl

Gültiger Bereich: Ganzzahlen zwischen 0 und 1.000.

Erforderlich: Nein

Der Gewichtungswert gibt den relativen Prozentsatz der Gesamtzahl der gestarteten Aufgaben an, die den angegebenen Kapazitätsanbieter verwenden.

Nehmen Sie beispielsweise an, Sie haben eine Strategie, die zwei Kapazitätsanbieter enthält und beide eine Gewichtung von einem haben. Wenn die Basis zufrieden ist, werden die Aufgaben gleichmäßig auf die beiden Kapazitätsanbieter aufgeteilt. Unter Verwendung derselben Logik nehmen Sie an, dass Sie für capacityProviderA ein Gewicht von 1 und für capacityProviderB ein Gewicht von 4 angeben. Dann wird für jede Aufgabe, die mit capacityProviderA ausgeführt wird, vier Aufgaben mit capacityProviderB ausgeführt.

`base`

Typ: Ganzzahl

Gültiger Bereich: Ganzzahlen zwischen 0 und 100.000.

Erforderlich: Nein

Der Basiswert gibt an, wie viele Aufgaben mindestens mit dem angegebenen Kapazitätsanbieter ausgeführt werden sollen. In einer Kapazitätsanbieterstrategie kann nur für einen Kapazitätsanbieter ein Basiswert festgelegt werden.

Aufgabendefinition

`taskDefinition`

Typ: Zeichenfolge

Erforderlich: Nein

`family` und `revision` (`family:revision`) oder der vollständige Amazon-Ressourcenname (ARN) der Aufgabendefinition, die in Ihrem Service ausgeführt werden soll. Wenn keine `revision` angegeben ist, wird die neueste `ACTIVE`-Revision der angegebenen Familie verwendet.

Eine Aufgabendefinition muss angegeben werden, wenn der Rolling-Update (`ECS`)-Bereitstellungs-Controller verwendet wird.

Plattform-Betriebssystem

`platformFamily`

Typ: Zeichenfolge

Required: Conditional

Standard: Linux

Dieser Parameter ist für Amazon-ECS-Services erforderlich, die auf Fargate gehostet werden.

Dieser Parameter wird für Amazon ECS-Services, die auf Amazon EC2 gehostet werden, ignoriert.

Das Betriebssystem auf den Containern, auf denen der Service ausgeführt wird. Die gültigen Werte sind `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` und `WINDOWS_SERVER_2022_CORE`.

Der `platformFamily`-Wert für jede Aufgabe, die Sie für den Service angeben, muss mit dem `platformFamily`-Wert des Services übereinstimmen. Zum Beispiel: Wenn Sie

`platformFamily` auf `WINDOWS_SERVER_2019_FULL` festlegen, muss der `platformFamily`-Wert für alle Aufgaben `WINDOWS_SERVER_2019_FULL` sein.

Plattformversion

`platformVersion`

Typ: Zeichenfolge

Erforderlich: Nein

Die Plattformversion, auf der Ihre Aufgaben im Service ausgeführt werden. Eine Plattformversion ist nur für Aufgaben mit dem Starttyp Fargate vorgesehen. Ist kein solcher angegeben, wird standardmäßig die neueste Version (`LATEST`) verwendet.

AWS Fargate-Plattformversionen werden verwendet, um auf eine bestimmte Laufzeitumgebung für die Fargate-Task-Infrastruktur zu verweisen. Bei der Angabe der `LATEST`-Plattformversion bei Ausführung einer Aufgabe oder beim Erstellen eines Service erhalten Sie die aktuellste Plattformversion, die für Ihre Aufgaben zur Verfügung steht. Wenn Sie Ihren Service skalieren, erhalten diese Aufgaben die Plattformversion, die in der aktuellen Bereitstellung des Service angegeben wurde. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

> **Note**
>
> Plattformversionen werden nicht für Aufgaben angegeben, die den Starttyp EC2 verwenden.

Cluster

`cluster`

Typ: Zeichenfolge

Erforderlich: Nein

Die Kurzbezeichnung oder der vollständige Amazon-Ressourcenname (ARN) des Clusters, auf dem Sie Ihren Service ausführen. Wenn Sie keinen Cluster angeben, wird der `default`-Cluster verwendet.

Service-Name

`serviceName`

Typ: Zeichenfolge

Erforderlich: Ja

Der Name Ihres Service. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Ziffern, Bindestriche und Unterstriche sind zulässig. Servicenamen in einem Cluster müssen eindeutig sein. Sie können jedoch ähnlich benannte Services in mehreren Clustern innerhalb einer Region oder in mehreren Regionen haben.

Einplanungsstrategie

`schedulingStrategy`

Typ: Zeichenfolge

Zulässige Werte: `REPLICA` | `DAEMON`

Erforderlich: Nein

Die Einplanungsstrategie, die verwendet werden soll. Wenn keine Einplanungsstrategie angegeben wird, wird die `REPLICA`-Strategie verwendet. Weitere Informationen finden Sie unter Amazon-ECS-Dienstleistungen.

Es gibt zwei Strategien für Service-Scheduler:

- `REPLICA`: Die Replica-Einplanungsstrategie platziert und die gewünschte Anzahl von Aufgaben in Ihrem Cluster und behält sie bei. Standardmäßig verteilt der Service-Scheduler Aufgaben über Availability Zones. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie festlegen, wie Aufgaben platziert und beendet werden. Weitere Informationen finden Sie unter Replikat-Strategie.
- `DAEMON`: Die Daemon-Einplanungsstrategie stellt genau eine Aufgabe auf jeder aktiven Container-Instance bereit, die alle von Ihnen in Ihrem Cluster angegebenen Platzierungsbedingungen für die Aufgaben erfüllt. Bei Verwendung dieser Strategie ist es nicht erforderlich, eine gewünschte Anzahl von Aufgaben oder eine Aufgabenplatzierungsstrategie anzugeben oder Auto-Scaling-Richtlinien zu verwenden. Weitere Informationen finden Sie unter Daemon-Strategie.

> **Note**
>
> Fargate-Aufgaben unterstützen die DAEMON-Einplanungsstrategie nicht.

Gewünschte Anzahl

`desiredCount`

Typ: Ganzzahl

Erforderlich: Nein

Die Anzahl der Instanziierungen der angegebenen Aufgabendefinition, die in Ihrem Service platziert und ausgeführt werden sollen.

Dieser Parameter ist erforderlich, wenn die REPLICA-Einplanungsstrategie verwendet wird. Wenn der Service die DAEMON-Einplanungsstrategie verwendet, ist dieser Parameter optional.

Bereitstellungskonfiguration

`deploymentConfiguration`

Typ: Objekt

Erforderlich: Nein

Optionale Bereitstellungsparameter zur Steuerung, wie viele Aufgaben während der Bereitstellung ausgeführt werden, und zur Steuerung der Reihenfolge beim Starten oder Stoppen von Aufgaben.

`maximumPercent`

Typ: Ganzzahl

Erforderlich: Nein

Wenn ein Service den Bereitstellungstyp der fortlaufenden Aktualisierung (ECS) verwendet, stellt der `maximumPercent`-Parameter eine Obergrenze für die Anzahl der Aufgaben Ihres Services dar, die während einer Bereitstellung im RUNNING-, STOPPING- oder PENDING-Status zulässig sind. Es wird als Prozentsatz des `desiredCount` ausgedrückt, der auf den nächsten ganzen Wert abgerundet wird. Mithilfe dieses Parameters können Sie die

Größe der Verteilungsstapel definieren. Wenn Ihr Service beispielsweise den Service-Scheduler `REPLICA` verwendet und einen `desiredCount`-Wert von vier Aufgaben und einen `maximumPercent`-Wert von 200 % hat, könnte der Scheduler vier neue Aufgaben starten, bevor er die vier älteren Aufgaben stoppt. Voraussetzung dafür ist, dass die dafür erforderlichen Cluster-Ressourcen zur Verfügung stehen. Der `maximumPercent`-Standardwert für einen Service mit dem `REPLICA`-Service-Scheduler beträgt 200 %.

Wenn Ihr Service den Service-Scheduler-Typ `DAEMON` verwendet, sollte `maximumPercent` bei 100 % verbleiben. Dies ist der Standardwert.

Die maximale Anzahl von Aufgaben während einer Bereitstellung ist die `desiredCount` multipliziert mit dem `maximumPercent`/100, abgerundet auf die nächste ganze Zahl.

Bei einem Service mit entweder dem Bereitstellungstyp „Blau/Grün“ (`CODE_DEPLOY`) oder `EXTERNAL` und bei Aufgaben mit dem Starttyp EC2 ist der Wert für den maximalen Prozentsatz auf den Standardwert eingestellt und wird zum Definieren der Obergrenze für die Anzahl der Aufgaben im Service verwendet, die im Zustand `RUNNING` verbleiben, während sich die Container-Instances im Zustand `DRAINING` befinden. Wenn die Aufgaben im Service den Starttyp Fargate verwenden, wird der maximale Prozentwert nicht verwendet, obwohl er bei der Beschreibung Ihres Services zurückgegeben wird.

`minimumHealthyPercent`

Typ: Ganzzahl

Erforderlich: Nein

Wenn ein Service den Bereitstellungstyp der fortlaufenden Aktualisierung (`ECS`) verwendet, stellt der `minimumHealthyPercent` eine Untergrenze für die Anzahl der Aufgaben Ihres Service dar, die während einer Bereitstellung im `RUNNING`-Status bleiben müssen. Dies wird als Prozentsatz des `desiredCount` ausgedrückt, der auf den nächsten ganzen Wert abgerundet wird. Sie können diesen Parameter verwenden, um die Bereitstellung ohne zusätzliche Cluster-Kapazität durchzuführen. Wenn Ihr Service beispielsweise eine `desiredCount` von vier Aufgaben und einen `minimumHealthyPercent` von 50 % hat, stoppt der Service-Scheduler zwei vorhandene Aufgaben, um Cluster-Kapazität freizugeben, bevor er zwei neue Aufgaben startet.

Bei Services, die keinen Load Balancer verwenden, ist Folgendes zu beachten:

- Ein Service gilt als fehlerfrei, wenn alle entscheidenden Container innerhalb der Aufgaben im Service ihre Zustandsprüfungen bestehen.

- Wenn für eine Aufgabe keine wesentlichen Container mit einer Zustandsprüfung definiert sind, wartet der Service-Scheduler 40 Sekunden, nachdem eine Aufgabe den `RUNNING`-Status erreicht hat, bevor die Aufgabe auf den minimalen fehlerfreien Gesamtprozentsatz angerechnet wird.
- Wenn für eine Aufgabe ein oder mehrere wesentliche Container mit einer Zustandsprüfung definiert sind, wartet der Service-Scheduler, bis die Aufgabe einen fehlerfreien Status erreicht, bevor er sie auf den minimalen fehlerfreien Gesamtprozentsatz anrechnet. Eine Aufgabe gilt als fehlerfrei, wenn alle entscheidenden Container innerhalb der Aufgabe ihre Zustandsprüfungen bestanden haben. Wie lange der Service-Scheduler warten kann, wird durch die Einstellungen für die Zustandsprüfung der Container bestimmt. Weitere Informationen finden Sie unter Zustandsprüfung.

Bei Services, die einen Load Balancer verwenden, ist Folgendes zu beachten:

- Wenn für eine Aufgabe keine wesentlichen Container mit einer Zustandsprüfung definiert sind, wartet der Service-Scheduler, bis die Zustandsprüfung der Load Balancer-Zielgruppe einen fehlerfreien Status zurückgibt, bevor er die Aufgabe auf den Mindestprozentsatz für den fehlerfreien Zustand anrechnet.
- Wenn für eine Aufgabe ein wesentlicher Container mit einer Zustandsprüfung definiert ist, wartet der Service-Scheduler darauf, dass sowohl die Aufgabe einen fehlerfreien Status erreicht als auch die Zustandsprüfung der Load-Balancer-Zielgruppe einen fehlerfreien Status zurückgibt, bevor er die Aufgabe auf den Mindestprozentsatz an fehlerfreien Aufgaben anrechnet.

Der Standardwert für einen Replica-Service für `minimumHealthyPercent` ist 100 %. Der `minimumHealthyPercent` Standardwert für einen Service, der den `DAEMON` Serviceplan verwendet, ist 0% für die AWS CLI, die AWS SDKs und die APIs und 50% für. AWS Management Console

Die Mindestanzahl fehlerfreier Aufgaben während einer Bereitstellung ist die `desiredCount` multipliziert mit dem `minimumHealthyPercent`/100, aufgerundet auf die nächste ganze Zahl.

Bei einem Service, der entweder Bereitstellungstyp „Blau/Grün“ (`CODE_DEPLOY`) oder `EXTERNAL` verwendet und Aufgaben mit dem Starttyp EC2 ausführt, ist der Wert für den mindestens fehlerfreier Prozentsatz auf den Standardwert eingestellt und wird zum Definieren der Untergrenze für die Anzahl der Aufgaben im Service verwendet, die im Zustand `RUNNING` verbleiben, während sich die Container-Instances im Zustand `DRAINING`

befinden. Wenn die Aufgaben im Service den Starttyp Fargate ausführen und der Service entweder Bereitstellungstyp „Blau/Grün“ (CODE_DEPLOY) oder EXTERNAL verwendet, wird der mindestens fehlerfreie Prozentwert nicht verwendet, obwohl er bei der Beschreibung Ihres Service zurückgegeben wird.

Bereitstellungs-Controller

`deploymentController`

Typ: Objekt

Erforderlich: Nein

Der für den Service zu verwendende Bereitstellungs-Controller. Wenn kein Bereitstellungs-Controller angegeben wird, wird der ECS-Controller verwendet. Weitere Informationen finden Sie unter Amazon-ECS-Dienstleistungen.

`type`

Typ: Zeichenfolge

Zulässige Werte: ECS | CODE_DEPLOY | EXTERNAL

Erforderlich: Ja

Der zu verwendende Controller-Typ der Bereitstellung. Es sind drei Bereitstellungs-Controller-Typen verfügbar:

ECS

Bei dem Bereitstellungstyp der fortlaufenden Aktualisierung (ECS) wird die aktuell ausgeführte Version des Containers durch die neueste Version ersetzt. Die Anzahl von Containern, die von Amazon ECS während einer fortlaufenden Aktualisierung zum Service hinzugefügt oder daraus entfernt werden, wird durch Anpassen der minimal und maximal zulässigen Anzahl fehlerfreier Aufgaben während einer Service-Bereitstellung wie unter deploymentConfiguration angegeben gesteuert.

CODE_DEPLOY

Der Bereitstellungstyp blau/grün (CODE_DEPLOY) verwendet das Bereitstellungsmodell blau/grün (powered by) CodeDeploy, mit dem Sie eine neue Bereitstellung eines Dienstes überprüfen können, bevor Sie Produktionsdatenverkehr an diesen senden.

`EXTERNAL`

Verwenden Sie den externen Bereitstellungstyp, wenn Sie einen beliebigen Drittanbieter-Bereitstellungs-Controller für die vollständige Kontrolle über den Bereitstellungsprozess für einen Amazon-ECS-Service verwenden möchten.

Platzierung von Aufgaben

`placementConstraints`

Typ: Array von -Objekten

Erforderlich: Nein

Ein Array von Platzierungseinschränkungsobjekten, die für Aufgaben in Ihrem Service verwendet werden sollen. Sie können maximal zehn Einschränkungen pro Aufgabe festlegen. Das Limit enthält Einschränkungen in der Aufgabendefinition und solche, die während der Laufzeit festgelegt werden. Wenn Sie den Starttyp Fargate verwenden, werden keine Einschränkungen für die Aufgabenplatzierung unterstützt.

`type`

Typ: Zeichenfolge

Erforderlich: Nein

Der Typ der Einschränkung. Um sicherzustellen, dass die einzelnen Aufgaben in einer bestimmten Gruppe auf einer anderen Container-Instance ausgeführt werden, verwenden Sie `distinctInstance`. Verwenden Sie `memberOf`, um die Auswahl auf eine Gruppe gültiger Kandidaten zu beschränken. Der Wert `distinctInstance` wird in Aufgabendefinitionen nicht unterstützt.

`expression`

Typ: Zeichenfolge

Erforderlich: Nein

Ein Ausdruck in Cluster-Abfragesprache, der auf die Einschränkung anzuwenden ist. Sie können keinen Ausdruck angeben, wenn der Einschränkungstyp `distinctInstance` lautet. Weitere Informationen finden Sie unter Erstellen Sie Ausdrücke, um Container-Instances für Amazon ECS-Aufgaben zu definieren.

`placementStrategy`

Typ: Array von -Objekten

Erforderlich: Nein

Die Platzierungsstrategieobjekte, die für Aufgaben in Ihrem Service verwendet werden sollen. Sie können maximal vier Strategieregeln pro Service festlegen.

`type`

Typ: Zeichenfolge

Zulässige Werte: `random` | `spread` | `binpack`

Erforderlich: Nein

Der Typ der Platzierungsstrategie. Die `random`-Platzierungsstrategie platziert zufällig Aufgaben in verfügbaren Kandidaten. Die `spread`-Platzierungsstrategie verteilt die Platzierungen anhand der `field`-Parameter gleichmäßig auf die verfügbaren Kandidaten. Die `binpack`-Strategie platziert Aufgaben in verfügbaren Kandidaten, die mindestens über die im `field`-Parameter festgelegte Ressourcenmenge verfügen. Wenn Sie beispielsweise den Arbeitsspeicher verringern, wird eine Aufgabe auf der Instance mit dem geringsten verbleibenden Arbeitsspeicher platziert, der aber noch ausreicht, um die Aufgabe auszuführen.

`field`

Typ: Zeichenfolge

Erforderlich: Nein

Das Feld zur Anwendung der Platzierungsstrategie. Gültige Werte für die `spread`-Platzierungsstrategie sind `instanceId` (oder `host` mit denselben Auswirkungen) oder jedes weitere Plattform- oder benutzerdefiniertes Attribut, das auf ein Container-Instance angewendet wird, z. B. `attribute:ecs.availability-zone`. Für die `binpack`-Placement-Strategie gültige Werte sind `cpu` und `memory`. Für die `random`-Placement-Strategie wird dieses Feld nicht verwendet.

Tags

`tags`

Typ: Array von -Objekten

Erforderlich: Nein

Die Metadaten, die Sie auf den Service anwenden, um die Kategorisierung und Organisation zu erleichtern. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen. Wenn ein Service gelöscht wird, werden auch die Tags gelöscht. Es können maximal 50 Tags auf den Service angewendet werden. Weitere Informationen finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.

`key`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 128 Zeichen.

Erforderlich: Nein

Ein Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Schlüssel ist eine allgemeine Bezeichnung, die wie eine Kategorie für spezifischere Tag-Werte fungiert.

`value`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

Der optionale Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Wert fungiert als Deskriptor in einer Tag-Kategorie (Schlüssel).

`enableECSManagedTags`

Typ: Boolesch

Zulässige Werte: `true` | `false`

Erforderlich: Nein

Gibt an, ob von Amazon ECS verwaltete Tags für Aufgaben im Service verwendet werden sollen. Der Standardwert ist `false`, wenn kein Wert angegeben wird. Weitere Informationen finden Sie unter Verwenden Sie Tags für die Abrechnung.

`propagateTags`

Typ: Zeichenfolge

Zulässige Werte: `TASK_DEFINITION` | `SERVICE`

Erforderlich: Nein

Gibt an, ob die Tags von der Aufgabendefinition oder dem Service in die Aufgaben in dem Service kopiert werden sollen. Wenn kein Wert angegeben wird, werden die Tags nicht kopiert. Tags können nur während der Serviceerstellung in die Aufgaben in dem Service kopiert werden. Wenn Sie Tags nach der Service- oder Aufgabenerstellung einer Aufgabe hinzufügen möchten, verwenden Sie die API-Aktion `TagResource`.

Netzwerkkonfiguration

`networkConfiguration`

Typ: Objekt

Erforderlich: Nein

Die Netzwerkkonfiguration für den Service. Dieser Parameter ist für Aufgabendefinitionen erforderlich, die den Netzwerkmodus `awsvpc` verwenden, um ihre eigene Elastic-Network-Schnittstelle zu empfangen. Für andere Netzwerkmodus wird er nicht unterstützt. Wenn Sie den Fargate-Starttyp verwenden, ist der Netzwerkmodus `awsvpc` erforderlich. Weitere Informationen über Netzwerke für den Amazon EC2 EC2-Starttyp finden Sie unterNetzwerkoptionen für Amazon ECS-Aufgaben für den EC2-Starttyp. Weitere Informationen zu Netzwerken für den Fargate-Starttyp finden Sie unter Fargate Task Networking.

`awsvpcConfiguration`

Typ: Objekt

Erforderlich: Nein

Ein Objekt, das die Subnetze und Sicherheitsgruppen für eine Aufgabe oder einen Service darstellt.

`subnets`

Typ: Zeichenfolgen-Array

Erforderlich: Ja

Die Subnetze, die der Aufgabe oder dem Service zugeordnet sind. Es gibt ein Limit von 16 Subnetzen, die gemäß `awsvpcConfiguration` festgelegt werden können.

`securityGroups`

Typ: Zeichenfolgen-Array

Erforderlich: Nein

Die Sicherheitsgruppen, die der Aufgabe oder dem Service zugeordnet sind. Wenn Sie keine Sicherheitsgruppe angeben, wird die Standardsicherheitsgruppe für die VPC verwendet. Es gibt ein Limit von fünf Sicherheitsgruppen, die basierend auf `awsvpcConfiguration` festgelegt werden können.

`assignPublicIP`

Typ: Zeichenfolge

Zulässige Werte: `ENABLED` | `DISABLED`

Erforderlich: Nein

Gibt an, ob die Elastic-Network-Schnittstelle eine öffentliche IP-Adresse erhält. Wenn kein Wert angegeben wird, wird der Standardwert `DISABLED` verwendet.

`healthCheckGracePeriodSeconds`

Typ: Ganzzahl

Erforderlich: Nein

Die Zeitdauer in Sekunden, die der Amazon-ECS-Service-Scheduler fehlerhafte Elastic Load Balancing-Zustandsprüfungen des Ziels, Container-Zustandsprüfungen und Route 53-Zustandsprüfungen ignorieren soll, nachdem eine Aufgabe in den Status `RUNNING` wechselt. Dies gilt nur, wenn Ihr Service für die Verwendung eines Load Balancers konfiguriert ist. Wenn für Ihren Service ein Load Balancer definiert ist und Sie keinen Wert für die Übergangsfrist der Zustandsprüfung angeben, wird der Standardwert von `0` verwendet.

Wenn es eine Weile dauert, bis die Aufgaben Ihres Service starten und auf die Zustandsprüfungen reagieren, können Sie für die Zustandsprüfungen eine Übergangsfrist von bis zu 2,147,483,647 Sekunden angeben. Während dieses Zeitraums ignoriert der ECS-Service-Scheduler den Status der Zustandsprüfungen. Durch diese Übergangsfrist wird verhindert, dass der ECS-Service-Scheduler Aufgaben als fehlerhaft markiert und stoppt, bevor sie gestartet werden können.

Wenn Sie kein Elastic Load Balancing verwenden, empfehlen wir Ihnen, die `startPeriod` in den Zustandsprüfungs-Parametern der Aufgabendefinition zu verwenden. Weitere Informationen finden Sie unter Ermitteln des Status von Amazon ECS-Aufgaben mithilfe von Container-Zustandsprüfungen.

`loadBalancers`

Typ: Array von -Objekten

Erforderlich: Nein

Ein Load Balancer-Objekt, das den Load Balancer angibt, der mit Ihrem Service verwendet werden soll. Für Services, die einen Application Load Balancer oder einen Network Load Balancer verwenden, gibt es ein Limit von fünf Zielgruppen, die Sie an einen Service anfügen können.

Nachdem Sie einen Dienst erstellt haben, kann die Load Balancer-Konfiguration von AWS Management Console nicht mehr geändert werden. Sie können den AWS Copilot AWS CLI oder das SDK verwenden AWS CloudFormation, um die Load Balancer-Konfiguration nur für den `ECS` Rolling Deployment Controller zu ändern, nicht für AWS CodeDeploy blau/grün oder extern. Wenn Sie eine Konfiguration für den Load Balancer hinzufügen, aktualisieren oder entfernen, startet Amazon ECS eine neue Bereitstellung mit der aktualisierten Konfiguration Elastic Load Balancing. Dies führt dazu, dass sich Aufgaben bei Load Balancern registrieren und von diesen abmelden. Wir empfehlen, dass Sie dies in einer Testumgebung überprüfen, bevor Sie die Konfiguration Elastic Load Balancing Balancing aktualisieren. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz.

Für Application Load Balancers und Network Load Balancers muss dieses Objekt den Zielgruppen-ARN des Load Balancers, den Namen des Containers (wie in der Containerdefinition angegeben) sowie den Container-Port enthalten, auf den vom Load Balancer aus zugegriffen werden soll. Wenn eine Aufgabe von diesem Service auf eine Container-Instance gesetzt wird, wird die Kombination aus Container-Instance und Port als Ziel in der hier angegebenen Zielgruppe registriert.

`targetGroupArn`

Typ: Zeichenfolge

Erforderlich: Nein

Der vollständige Amazon-Ressourcenname (ARN) der Elastic-Load-Balancing-Zielgruppe die einem Service angefügt sind.

Ein Zielgruppen-ARN wird nur bei Verwendung eines Application Load Balancers oder eines Network Load Balancers angegeben.

`loadBalancerName`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Load Balancer, der dem Service zugeordnet werden soll.

Wenn Sie einen Application Load Balancer oder einen Network Load Balancer verwenden, lassen Sie den Load-Balancer-Name-Parameter weg.

`containerName`

Typ: Zeichenfolge

Erforderlich: Nein

Der Name des Containers (wie in der Containerdefinition angegeben), der mit dem Load Balancer verknüpft werden soll.

`containerPort`

Typ: Ganzzahl

Erforderlich: Nein

Der Port auf dem Container, der mit dem Load Balancer verknüpft werden soll. Dieser Port muss einem `containerPort` in der Aufgabendefinition entsprechen, die von den Aufgaben im Service verwendet wird. Für Aufgaben, die den EC2-Starttyp verwenden, muss die Container Instance eingehenden Datenverkehr auf dem `hostPort` der Port-Zuweisung zulassen.

`role`

Typ: Zeichenfolge

Erforderlich: Nein

Der Kurzname oder vollständige ARM der IAM-Rolle, die es Amazon ECS ermöglicht, Aufrufe in Ihrem Namen an Ihren Load Balancer vorzunehmen. Dieser Parameter ist nur zulässig, wenn Sie einen Load Balancer mit einer einzelnen Zielgruppe für Ihren Service verwenden und Ihre

Aufgabendefinition nicht den Netzwerkmodus `awsvpc` verwendet. Wenn Sie den `role`-Parameter festlegen, müssen Sie auch ein Load Balancer-Objekt mit dem `loadBalancers`-Parameter angeben.

Wenn Ihre festgelegte Rolle einen anderen Pfad als `/` aufweist, müssen Sie entweder den vollständigen Rollen-ARN angeben (empfohlen) oder ein Präfix mit dem Pfad vor dem Rollennamen verwenden. Wenn beispielsweise ein Rolle mit dem Namen `bar` den Pfad `/foo/` aufweist, würden Sie `/foo/bar` als Rollennamen angeben. Weitere Informationen finden Sie unter Anzeigenamen und Pfade im IAM-Benutzerhandbuch.

> **Important**
>
> Wenn Ihr Konto bereits über die servicegebundene Rolle für den Amazon ECS Service verfügt, wird diese Rolle standardmäßig für Ihren Service verwendet, es sei denn, Sie geben hier eine Rolle an. Die servicegebundene Rolle ist erforderlich, wenn Ihre Aufgabendefinition den Netzwerkmodus awsvpc verwendet. In diesem Fall sollten Sie hier keine Rolle angeben. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.

`serviceConnectConfiguration`

Typ: Objekt

Erforderlich: Nein

Die Konfiguration für diesen Service, um Services zu erkennen und eine Verbindung zu ihnen herzustellen sowie von anderen Services innerhalb eines Namespace erkannt und verbunden zu werden.

Weitere Informationen finden Sie unter Verwenden Sie Service Connect, um Amazon ECS-Services mit Kurznamen zu verbinden.

`enabled`

Typ: Boolesch

Erforderlich: Ja

Gibt an, ob Service Connect mit diesem Service verwendet werden soll.

`namespace`

Typ: Zeichenfolge

Erforderlich: Nein

Der Kurzname oder der vollständige Amazon Resource Name (ARN) des AWS Cloud Map Namespaces zur Verwendung mit Service Connect. Der Namespace muss sich in derselben AWS-Region befinden wie der Amazon-ECS-Service und der Cluster. Der Typ des Namespace hat keine Auswirkungen auf Service Connect. Weitere Informationen zu finden Sie AWS Cloud Map unter Working with Services im AWS Cloud Map Developer Guide.

`services`

Typ: Array von -Objekten

Erforderlich: Nein

Ein Array von Service-Connect-Serviceobjekten. Dies sind Namen und Aliase (auch als Endpunkte bezeichnet), die von anderen Amazon-ECS-Services verwendet werden, um eine Verbindung zu diesem Service herzustellen.

Dieses Feld ist für einen „Client“-Amazon-ECS-Service, der nur Mitglied eines Namespace ist, nicht erforderlich, um eine Verbindung zu anderen Services innerhalb des Namespace herzustellen. Ein Beispiel ist eine Frontend-Anwendung, die eingehende Anfragen entweder von einem Load Balancer, der dem Service angefügt ist, oder auf andere Weise akzeptiert.

Ein Objekt wählt einen Port aus der Aufgabendefinition aus, weist dem AWS Cloud Map Dienst einen Namen und eine Reihe von Aliasen (auch als Endpunkte bezeichnet) und Ports zu, über die Client-Anwendungen auf diesen Dienst verweisen können.

`portName`

Typ: Zeichenfolge

Erforderlich: Ja

Der `portName` muss mit dem `name` einer der `portMappings` aus allen Containern in der Aufgabendefinition dieses Amazon-ECS-Service übereinstimmen.

`discoveryName`

Typ: Zeichenfolge

Erforderlich: Nein

Das `discoveryName` ist der Name des neuen AWS Cloud Map Service, den Amazon ECS für diesen Amazon ECS-Service erstellt. Dieser muss innerhalb des AWS Cloud Map -Namespace eindeutig sein.

Wenn dieses Feld nicht angegeben ist, wird `portName` verwendet.

`clientAliases`

Typ: Array von -Objekten

Erforderlich: Nein

Die Liste der Client-Aliase für diesen Service-Connect-Service. Sie verwenden diese, um Namen zuzuweisen, die von Client-Anwendungen verwendet werden können. Die maximale Anzahl von Client-Aliasen, die Sie in dieser Liste haben können, ist 1.

Jeder Alias („Endpunkt“) ist ein DNS-Name und eine Portnummer, die andere Amazon-ECS-Services („Clients“) verwenden können, um eine Verbindung zu diesem Service herzustellen.

Jede Kombination aus Name und Port muss innerhalb des Namespace eindeutig sein.

Diese Namen werden innerhalb jeder Aufgabe des Client-Services konfiguriert, nicht in AWS Cloud Map. DNS-Anfragen zum Auflösen dieser Namen verlassen die Aufgabe nicht und werden nicht auf das Kontingent von DNS-Anfragen pro Sekunde pro Schnittstelle für elastische Netzwerke angerechnet.

`port`

Typ: Ganzzahl

Erforderlich: Ja

Die Überwachungsportnummer für den Service-Connect-Proxy. Dieser Port ist in allen Aufgaben innerhalb desselben Namespace verfügbar.

Um zu vermeiden, dass Ihre Anwendungen in Amazon-ECS-Client-Services geändert werden, stellen Sie diesen auf denselben Port ein, den die Client-Anwendung standardmäßig verwendet.

`dnsName`

Typ: Zeichenfolge

Erforderlich: Nein

Der `dnsName` ist der Name, den Sie in den Anwendungen von Client-Aufgaben verwenden, um eine Verbindung zu diesem Service herzustellen. Der Name muss eine gültige DNS-Kennzeichnung sein.

Wenn dieses Feld nicht angegeben ist, ist der Standardwert das `discoveryName.namespace`. Wenn der `discoveryName` nicht angegeben ist, wird der `portName` aus der Aufgabendefinition verwendet.

Um zu vermeiden, dass Ihre Anwendungen in Amazon-ECS-Client-Services geändert werden, stellen Sie diesen auf denselben Namen ein, den die Client-Anwendung standardmäßig verwendet. Einige gängige Namen sind beispielsweise `database`, `db` oder der Name einer Datenbank in Kleinbuchstaben, wie z. B. `mysql` oder `redis`.

`ingressPortOverride`

Typ: Ganzzahl

Erforderlich: Nein

(Optional) Die Portnummer, die der Service-Connect-Proxy überwachen soll.

Verwenden Sie den Wert dieses Feldes, um den Proxy für den Datenverkehr an der Portnummer zu umgehen, die in der Aufgabendefinition dieser Anwendung unter dem Namen `portMapping` angegeben ist. Verwenden Sie diesen dann in Ihren Amazon-VPC-Sicherheitsgruppen, um den Datenverkehr in den Proxy für diesen Amazon-ECS-Service zuzulassen.

Im `awsvpc`-Modus ist der Standardwert die Container-Portnummer, die in der Aufgabendefinition dieser Anwendung im benannten `portMapping` angegeben ist. Im `bridge`-Modus ist der Standardwert der dynamische flüchtige Port des Service-Connect-Proxys.

`logConfiguration`

Typ: LogConfigurationObjekt

Erforderlich: Nein

Dies definiert, wo die Service-Connect-Proxyprotokolle veröffentlicht werden. Verwenden Sie die Protokolle zum Debuggen bei unerwarteten Ereignissen. Diese Konfiguration legt den `logConfiguration`-Parameter im Service-Connect-Proxy-Container in jeder Aufgabe in diesem Amazon-ECS-Service fest. Der Proxy-Container ist nicht in der Aufgabendefinition angegeben.

Wir empfehlen, dieselbe Protokollkonfiguration wie die Anwendungscontainer der Aufgabendefinition für diesen Amazon-ECS-Service zu verwenden. Für FireLens ist dies die Protokollkonfiguration des Anwendungscontainers. Es ist nicht der FireLens Log-Router-Container, der das `fluentd` Container-Image `fluent-bit` oder verwendet.

`serviceRegistries`

Typ: Array von -Objekten

Erforderlich: Nein

Die Details der Serviceerkennungskonfiguration für Ihren Service. Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.

`registryArn`

Typ: Zeichenfolge

Erforderlich: Nein

Der Amazon-Ressourcenname (ARN) der Serviceregistrierung. Die derzeit unterstützte Dienstregistrierung ist AWS Cloud Map. Weitere Informationen finden Sie unter Arbeiten mit Services im AWS Cloud Map -Entwicklerhandbuch.

`port`

Typ: Ganzzahl

Erforderlich: Nein

Der Port-Wert, der verwendet wird, wenn Ihr Service zur Service Discovery einen SRV-Datensatz angegeben hat. Dieses Feld ist erforderlich, wenn der `awsvpc`-Netzwerkmodus und SRV-Datensätze verwendet werden.

`containerName`

Typ: Zeichenfolge

Erforderlich: Nein

Der Containernamen-Wert, der für Ihren Service zur Service Discovery verwendet werden soll. Dieser Wert wird in der Aufgabendefinition festgelegt. Wenn die Aufgabendefinition, die Ihre Serviceaufgabe angibt, den Netzwerkmodus `bridge` oder `host` verwendet, müssen Sie eine Kombination aus `containerName` und `containerPort` aus der Aufgabendefinition angeben. Wenn die Aufgabendefinition, die Ihre Serviceaufgabe angibt, den Netzwerkmodus `awsvpc` verwendet und ein SRV-DNS-Datensatz verwendet wird, müssen Sie entweder eine Kombination aus `containerName` und `containerPort` angeben, oder einen `port`-Wert, aber nicht beides.

`containerPort`

Typ: Ganzzahl

Erforderlich: Nein

Der Port-Wert, der für Ihren Service zur Service Discovery verwendet werden soll. Dieser Wert wird in der Aufgabendefinition festgelegt. Wenn die Aufgabendefinition, die Ihre Serviceaufgabe angibt, den Netzwerkmodus `bridge` oder `host` verwendet, müssen Sie eine Kombination aus `containerName` und `containerPort` aus der Aufgabendefinition angeben. Wenn die Aufgabendefinition, die Ihre Serviceaufgabe angibt, den Netzwerkmodus `awsvpc` verwendet und ein SRV-DNS-Datensatz verwendet wird, müssen Sie entweder eine Kombination aus `containerName` und `containerPort` angeben, oder einen `port`-Wert, aber nicht beides.

Client-Token

`clientToken`

Typ: Zeichenfolge

Erforderlich: Nein

Der eindeutige Bezeichner, bei dem die Groß- und Kleinschreibung beachtet werden muss, um die Idempotenz der Anfrage sicherzustellen. Er kann bis zu 32 ASCII-Zeichen lang sein.

Volumenkonfigurationen

`volumeConfigurations`

Typ: Objekt

Erforderlich: Nein

Die Konfiguration, die verwendet wird, um Volumes für Aufgaben zu erstellen, die vom Dienst verwaltet werden. Für jede Aufgabe im Service wird ein Volume erstellt. Nur Volumes, die `configuredAtLaunch` in der Aufgabendefinition als gekennzeichnet sind, können mithilfe dieses Objekts konfiguriert werden. Dieses Objekt ist erforderlich, um Amazon EBS-Datenvolumen an Aufgaben anzuhängen, die von einem Service verwaltet werden. Weitere Informationen finden Sie unter Amazon EBS-Volumes.

`name`

Typ: Zeichenfolge

Erforderlich: Ja

Der Name eines Volumes, das bei der Erstellung oder Aktualisierung eines Service konfiguriert wurde. Bis zu 255 Buchstaben (Groß- und Kleinbuchstaben), Zahlen, Unterstriche (_) und Bindestriche () sind zulässig. - Dieser Wert muss mit dem Volumennamen übereinstimmen, der in der Aufgabendefinition angegeben ist.

`managedEBSVolume`

Typ: Objekt

Erforderlich: Nein

Die Volume-Konfiguration für Amazon EBS-Volumes, die Aufgaben zugewiesen werden, die von einem Service verwaltet werden, wenn ein Service erstellt oder aktualisiert wird.

`encrypted`

Typ: Boolesch

Erforderlich: Nein

Gültige Werte: `true|false`

Gibt an, ob das Amazon EBS-Volume, das an Aufgaben angehängt ist, die von einem Service verwaltet werden, verschlüsselt wird. Wenn Sie die Amazon EBS-Verschlüsselung

standardmäßig für Ihr Konto aktiviert haben, wird diese Einstellung überschrieben und das Volume wird verschlüsselt. Weitere Informationen zur standardmäßigen EBS-Verschlüsselung finden Sie unter Standardverschlüsselung im Amazon EC2 EC2-Benutzerhandbuch.

`kmsKeyId`

Typ: Zeichenfolge

Erforderlich: Nein

Die Kennung des AWS Key Management Service (AWS KMS) -Schlüssels, der für die Amazon EBS-Verschlüsselung verwendet werden soll. Wenn dieser Parameter nicht angegeben ist, wird Ihr AWS KMS key für Amazon EBS verwendet. Wenn `KmsKeyId` angegeben ist, muss der verschlüsselte Status `true` sein.

Sie können den KMS-Schlüssel mit einer der folgenden Methoden angeben:

- Schlüssel-ID — Zum Beispiel`1234abcd-12ab-34cd-56ef-1234567890ab`.
- Schlüsselalias — Zum Beispiel`alias/ExampleAlias`.
- Schlüssel-ARN — Zum Beispiel`arn:aws:kms:us-east-1:012345678910:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias ARN — Zum Beispiel`arn:aws:kms:us-east-1:012345678910:alias/ExampleAlias`.

> **Important**
>
> AWS authentifiziert den KMS-Schlüssel asynchron. Wenn Sie also eine ID, einen Alias oder einen ARN angeben, die nicht gültig sind, scheint die Aktion erfolgreich zu sein, schlägt aber letztendlich fehl. Weitere Informationen finden Sie unter Problembehandlung bei Amazon EBS-Volumenanhängen.

`volumeType`

Typ: Zeichenfolge

Erforderlich: Nein

Gültige Werte: `gp2` | `gp3` | `io1` | `io2` | `sc1` | `st1` `standard`

Der EBS-Volume-Typ. Weitere Informationen zu Volumetypen finden Sie unter Amazon EBS-Volumetypen im Amazon EC2 EC2-Benutzerhandbuch. Der Standard-Volume-Typ ist `gp3`.

> **Note**
>
> Der `standard` Volumetyp wird nicht für Amazon EBS-Volumes unterstützt, die für das Anhängen an Fargate-Aufgaben konfiguriert sind.

`sizeInGiB`

Typ: Ganzzahl

Erforderlich: Nein

Gültiger Bereich: Ganzzahlen zwischen 1 und 16.384

Die Größe des EBS-Volumes in Gibibyte (GiB). Wenn Sie keine Snapshot-ID angeben, um ein Volume für das Anhängen zu konfigurieren, müssen Sie einen Größenwert angeben. Wenn Sie ein Volume für das Anhängen mithilfe eines Snapshots konfigurieren, ist der Standardwert die Snapshot-Größe. Sie können dann eine Größe angeben, die größer oder gleich der Snapshot-Größe ist.

Für `gp3` Datenträgertypen `gp2` und -datenträgertypen liegt der gültige Bereich zwischen 1 und 16.384.

Für Datenträgertypen `io1` und `io2` -volumetypen liegt der gültige Bereich zwischen 4 und 16.384.

Für Datenträgertypen `st1` und `sc1` -volumetypen liegt der gültige Bereich zwischen 125 und 16.384.

Für den `standard` Datenträgertyp liegt der gültige Bereich zwischen 1 und 1.024.

`snapshotId`

Typ: Zeichenfolge

Erforderlich: Nein

Die ID des Snapshots eines vorhandenen EBS-Volumes, der verwendet wird, um ein neues Volume zu erstellen, das an die ECS-Aufgabe angehängt ist.

`iops`

Typ: Ganzzahl

Erforderlich: Nein

Die Anzahl der E/A-Vorgänge pro Sekunde (IOPS). Im Fall von `gp3`-, `io1`- und `io2`-Volumes stellt dieser Wert die Anzahl der IOPS dar, die für das Volume bereitgestellt werden. Bei `gp2` Volumes steht dieser Wert für die Ausgangsleistung des Volumes und für die Geschwindigkeit, mit der das Volume I/O-Credits für das Bursting sammelt. Dieser Parameter ist für `io1`- und `io2`-Volumes erforderlich. Dieser Parameter wird für`gp2`,`st1`, `sc1` oder `standard` Volumes nicht unterstützt.

Für `gp3` Volumen liegt der gültige Wertebereich zwischen 3.000 und 16.000.

Für `io1` Volumen liegt der gültige Wertebereich zwischen 100 und 64.000.

Für `io2` Volumen liegt der gültige Wertebereich zwischen 100 und 64.000.

`throughput`

Typ: Ganzzahl

Erforderlich: Nein

Der bereitzustellende Durchsatz für Volumes, die mit Aufgaben verknüpft sind, die von einem Dienst verwaltet werden.

> **Important**
>
> Dieser Parameter wird nur für `gp3` Volumes unterstützt.

`roleArn`

Typ: Zeichenfolge

Erforderlich: Ja

Der Amazon Resource ARN (ARN) der Infrastrukturrolle AWS Identity and Access Management (IAM), die Amazon ECS-Berechtigungen zur Verwaltung von Amazon EBS-Ressourcen für Ihre Aufgaben bereitstellt. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon ECS-Infrastruktur.

`tagSpecifications`

Typ: Objekt

Erforderlich: Nein

Die Spezifikation für Tags, die auf serviceverwaltete Amazon EBS-Volumes angewendet werden sollen.

`resourceType`

Typ: Zeichenfolge

Erforderlich: Ja

Zulässige Werte: `volume`

Der Typ der Ressource, die bei der Erstellung markiert werden soll.

`tags`

Typ: Array von -Objekten

Erforderlich: Nein

Die Metadaten, die Sie auf Bände anwenden, um sie zu kategorisieren und zu organisieren. Jedes Tag besteht aus einem Schlüssel und einem optionalen Wert, die Sie beide definieren. `AmazonECSCreated`und `AmazonECSManaged` sind reservierte Tags, die von Amazon ECS in Ihrem Namen hinzugefügt wurden, sodass Sie maximal 48 eigene Tags angeben können. Wenn ein Volume gelöscht wird, werden auch die Tags gelöscht. Weitere Informationen finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.

`key`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge beträgt 1 Zeichen. Maximale Länge beträgt 128 Zeichen.

Erforderlich: Nein

Ein Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Schlüssel ist eine allgemeine Bezeichnung, die wie eine Kategorie für spezifischere Tag-Werte fungiert.

`value`

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge beträgt 256 Zeichen.

Erforderlich: Nein

Der optionale Teil eines Schlüssel-Wert-Paares, aus dem ein Tag besteht. Ein Wert fungiert als Deskriptor in einer Tag-Kategorie (Schlüssel).

`propagateTags`

Typ: Zeichenfolge

Zulässige Werte: `TASK_DEFINITION` | `SERVICE` | `NONE`

Erforderlich: Nein

Gibt an, ob die Tags aus der Aufgabendefinition oder dem Dienst auf ein Volume kopiert werden sollen. Wenn `NONE` angegeben oder kein Wert angegeben ist, werden die Tags nicht kopiert.

`fileSystemType`

Typ: Zeichenfolge

Erforderlich: Nein

Zulässige Werte: `xfs|ext3|ext4`

Der Typ des Dateisystems auf einem Volume. Der Dateisystemtyp des Volumes bestimmt, wie Daten auf dem Volume gespeichert und abgerufen werden. Für Volumes, die aus einem Snapshot erstellt wurden, müssen Sie denselben Dateisystemtyp angeben, den das Volume bei der Erstellung des Snapshots verwendet hat. Wenn der Dateisystemtyp nicht übereinstimmt, kann die Aufgabe nicht gestartet werden. Die Standardeinstellung für Volumes, die an Linux-Aufgaben angehängt sind, ist. `XFS`

Servicedefinitionsvorlage

Im Folgenden ist die JSON-Darstellung einer Amazon-ECS-Servicedefinition dargestellt.

Amazon EC2 EC2-Starttyp

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
```

```
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "serviceConnectConfiguration": {
        "enabled": true,
```

```
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
```

```
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Fargate Starttyp

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "FARGATE",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
```

```
        }
    ],
    "platformVersion": "",
    "platformFamily": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
```

```
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
```

```
                "snapshotId": "",
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Sie können diese Servicedefinitionsvorlage mit dem folgenden AWS CLI -Befehl erstellen.

```
aws ecs create-service --generate-cli-skeleton
```

Verschlagwortung von Amazon ECS-Ressourcen

Um sich die Verwaltung Ihrer Amazon-ECS-Ressourcen zu erleichtern, können Sie optional jeder Ressource mithilfe von Tags Ihre eigenen Metadaten zuweisen. Jedes Tag besteht aus einem Schlüssel und einem optionalen Wert.

Mit Tags können Sie Ihre Amazon-ECS-Ressourcen auf unterschiedliche Weise kategorisieren (z. B. nach Zweck, Eigentümer oder Umgebung). Dies ist nützlich, wenn Sie viele Ressourcen desselben Typs haben. Sie können eine bestimmte Ressource anhand der ihr zugewiesenen Tags schnell identifizieren. Sie können zum Beispiel eine Reihe von Tags für die Amazon-ECS-Container-Instances Ihres Kontos definieren. Auf diese Weise können Sie den Eigentümer und die Stack-Ebene der einzelnen Instances nachverfolgen.

Sie können Tags für Ihre Kosten- und Nutzungsberichte verwenden. Sie können diese Berichte verwenden, um die Kosten und Nutzung Ihrer Amazon-ECS-Ressourcen analysieren. Weitere Informationen finden Sie unter the section called “Nutzungsberichte”.

> **Warning**
>
> Es gibt viele APIs, die Tag-Schlüssel und ihre Werte zurückgeben. Die Zugriffsverweigerung von `DescribeTags` verweigert nicht automtisch den Zugriff auf Tags (Markierungen), die von anderen APIs zurückgegeben wurden. Als bewährte Vorgehensweise empfehlen wir Ihnen, keine sensiblen Daten in Ihre Tags (Markierungen) aufzunehmen.

Wir empfehlen die Verwendung von Tag (Markierung)-Schlüsseln, die die Anforderungen der jeweiligen Ressourcentypen erfüllen. Eine Anzahl einheitlicher Tagschlüssel vereinfacht das Verwalten Ihrer Ressourcen. Sie können die Ressourcen auf Grundlage der hinzugefügten Tags (Markierungen) filtern und danach suchen.

Tags (Markierungen) haben keine semantische Bedeutung für Amazon ECS und werden ausschließlich als Zeichenfolgen interpretiert. Sie können Tag (Markierung)-Schlüssel und -Werte bearbeiten und Tags (Markierungen) jederzeit von einer Ressource entfernen. Sie können den Wert eines Tags (Markierung) zwar auf eine leere Zeichenfolge, jedoch nicht null festlegen. Wenn Sie ein Tag (Markierung) mit demselben Schlüssel wie ein vorhandener Tag (Markierung) für die Ressource hinzufügen, wird der alte Wert mit dem neuen überschrieben. Wenn Sie eine Ressource löschen, werden auch alle Tags für die Ressource gelöscht.

Wenn Sie AWS Identity and Access Management (IAM) verwenden, können Sie steuern, welche Benutzer in Ihrem AWS Konto berechtigt sind, Tags zu verwalten.

So werden Ressourcen markiert

Es gibt mehrere Möglichkeiten, Amazon-ECS-Aufgaben, -Services, -Aufgabendefinitionen und -Cluster zu markieren:

- Ein Benutzer markiert eine Ressource manuell mithilfe der AWS Management Console Amazon ECS-API AWS CLI, des oder eines AWS SDK.
- Ein Benutzer erstellt einen Service oder führt eine eigenständige Aufgabe aus und wählt die Option Amazon-ECS-verwaltete Tags aus.

  Amazon ECS markiert automatisch alle neu gestarteten Aufgaben. Weitere Informationen finden Sie unter the section called “Von Amazon ECS verwaltete Tags”.
- Ein Benutzer erstellt mithilfe der Konsole eine Ressource. Die Konsole markiert die Ressourcen automatisch.

  Diese Tags werden in den AWS CLI und AWS SDK-Antworten zurückgegeben und in der Konsole angezeigt. Sie können diese Tags nicht ändern oder löschen.

  Informationen zu den hinzugefügten Tags finden Sie in der Spalte Tags, die automatisch von der Konsole hinzugefügt wurden in der Tabelle Tagging-Unterstützung für Amazon-ECS-Ressourcen.

Wenn Sie bei der Erstellung einer Ressource Tags angeben und die Tags nicht angewendet werden können, macht Amazon ECS den Erstellungsprozess rückgängig. Auf diese Weise werden Ressourcen entweder mit Tags (Markierungen) oder überhaupt nicht erstellt und keine Ressourcen verbleiben ohne Tags (Markierungen). Indem Sie Ressourcen zum Erstellungszeitpunkt markieren, müssen Sie anschließend keine benutzerdefinierten Markierungs-Skripts ausführen.

Die folgende Tabelle beschreibt die Amazon-ECS-Ressourcen, die Tagging unterstützen.

Markierungsunterstützung für Amazon-ECS-Ressourcen

Ressource	Unterstützt Tags (Markierungen)	Unterstützt Tag-Propagierung	Von der Konsole automatisch hinzugefügte Tags
Amazon-ECS-Aufgaben	Ja	Ja, von der Aufgabendefinition.	Schlüssel: `aws:ecs:clusterName` Value (Wert): `cluster-name`
Amazon-ECS-Services	Ja	Ja, von der Aufgabendefinition oder dem Service zu den Aufgaben im Service.	Schlüssel: `ecs:service:stackId` Wert `arn:aws:cloudformation: arn`
Amazon-ECS-Aufgabensätze	Ja	Nein	N/A
Amazon-ECS-Aufgabendefinitionen	Ja	Nein	Schlüssel: `ecs:taskDefinition:createdFrom` Value (Wert): `ecs-console-v2`
Amazon-ECS-Cluster	Ja	Nein	Schlüssel: `aws:cloudformation:logical-id` Value (Wert): `ECSCluster`

Ressource	Unterstützt Tags (Markierungen)	Unterstützt Tag-Propagierung	Von der Konsole automatisch hinzugefügte Tags
			Schlüssel: `aws:cloudformation:stack-id` Value (Wert): `arn:aws:cloudformation:` *`arn`* Schlüssel: `aws:cloudformation:stack-name` Value (Wert): `ECS-Console-V2-Cluster-` *`EXAMPLE`*
Amazon-ECS-Container-Instances	Ja	Ja, von der Amazon-EC2-Instance. Weitere Informationen finden Sie unter Hinzufügen von Tags zu einer Amazon ECS-Container-Instance.	N/A
Externe Amazon-ECS-Instances	Ja	Nein	N/A

Ressource	Unterstützt Tags (Markierungen)	Unterstützt Tag-Propagierung	Von der Konsole automatisch hinzugefügte Tags
Amazon-ECS-Kapazitätsanbieter	Ja. Vordefinierte `FARGATE`- und `FARGATE_SPOT` -Kapazitätsanbieter können nicht markiert werden.	Nein	N/A

Markieren von Ressourcen bei der Erstellung

Die folgenden Ressourcen unterstützen das Tagging bei der Erstellung mithilfe der Amazon ECS-API oder des AWS SDK: AWS CLI

- Amazon-ECS-Aufgaben
- Amazon-ECS-Dienstleistungen
- Amazon ECS-Aufgabendefinition
- Amazon-ECS-Aufgabensätze
- Amazon-ECS-Cluster
- Amazon-ECS-Container-Instances
- Amazon-ECS-Kapazitätsanbieter

Amazon ECS bietet die Möglichkeit, die Tagging-Autorisierung für die Erstellung von Ressourcen zu verwenden. Wenn das für die Tagging-Autorisierung konfiguriert AWS-Konto ist, müssen Benutzer über Berechtigungen für Aktionen verfügen, mit denen die Ressource erstellt wird, z. B. `ecsCreateCluster` Wenn Sie bei der Aktion zur Erstellung der Ressource Tags angeben, AWS führt eine zusätzliche Autorisierung durch, um zu überprüfen, ob Benutzer oder Rollen berechtigt sind, Tags zu erstellen. Daher müssen Sie explizite Berechtigungen für die Verwendung der Aktion `ecs:TagResource` gewähren. Weitere Informationen finden Sie unter the section called “Zuordnung von Tags (Markierungen) zu Ressourcen während der Erstellung”. Informationen zur Konfiguration der Option finden Sie unter. the section called “Tagging-Autorisierung”

Einschränkungen

Für Tags gelten die folgenden Einschränkungen:

- Maximal 50 Tags können einer Ressource zugeordnet werden.
- Tag-Schlüssel können nicht für eine Ressource wiederholt werden. Jeder Tag-Schlüssel muss eindeutig sein, und jeder Schlüssel darf nur einen Wert besitzen.
- Schlüssel können aus bis zu 128 Zeichen im UTF-8-Format bestehen.
- Werte können aus bis zu 256 Zeichen im UTF-8-Format bestehen.
- Wenn mehrere AWS-Services und Ressourcen Ihr Tagging-Schema verwenden, beschränken Sie die Anzahl der verwendeten Zeichentypen. Einige Services haben möglicherweise Einschränkungen für zulässige Zeichen. Allgemein erlaubte Zeichen sind: Buchstaben, Zahlen, Leerzeichen und die folgenden Sonderzeichen: **+ - = . _ : / @**.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden.
- Verwenden Sie weder `aws:` noch `AWS:` oder Kombinationen aus Groß- und Kleinbuchstaben von diesen als Präfix für Schlüssel oder Werte, da sie für die -Verwendung reserviert sind. Diese sind nur für die AWS Verwendung reserviert. Sie können keine Tag-Schlüssel oder -Werte mit diesem Präfix bearbeiten oder löschen. Tags mit diesem Präfix werden nicht auf dein tags-per-resource Limit angerechnet.

Von Amazon ECS verwaltete Tags

Wenn Sie von Amazon ECS verwaltete Tags verwenden, kennzeichnet Amazon ECS automatisch alle neu gestarteten Aufgaben und alle Amazon EBS-Volumes, die mit den Aufgaben verknüpft sind, mit den Clusterinformationen und entweder den vom Benutzer hinzugefügten Aufgabendefinitions-Tags oder den Service-Tags. Im Folgenden werden die hinzugefügten Tags beschrieben:

- Eigenständige Aufgaben – ein Tag mit einem Schlüssel als `aws:ecs:clusterName` und einem Wert, der dem Clusternamen zugewiesen ist. Alle Aufgabendefinitions-Tags, die von Benutzern hinzugefügt wurden. Ein Amazon EBS-Volume, das an eine eigenständige Aufgabe angehängt ist, erhält das Tag mit einem Schlüssel als `aws:ecs:clusterName` und einem Wert, der auf den Clusternamen gesetzt ist. Weitere Informationen zum Amazon EBS-Volume-Tagging finden Sie unter Tagging Amazon EBS-Volumes.
- Aufgaben, die Teil eines Services sind – ein Tag mit einem Schlüssel als `aws:ecs:clusterName` und einem Wert, der dem Clusternamen zugewiesen ist. Ein Tag mit einem Schlüssel als

`aws:ecs:serviceName` und einem Wert, der auf den Servicenamen gesetzt ist. Tags aus einer der folgenden Ressourcen:

- Aufgabendefinitionen – Alle Aufgabendefinitions-Tags, die von Benutzern hinzugefügt wurden.
- Services – Alle Service-Tags, die von Benutzern hinzugefügt wurden.

  Ein Amazon EBS-Volume, das an eine Aufgabe angehängt ist, die Teil eines Service ist, erhält ein Tag mit einem Schlüssel als `aws:ecs:clusterName` und einem Wert, der auf den Clusternamen festgelegt ist, sowie ein Tag mit einem Schlüssel als `aws:ecs:serviceName` und einem Wert, der auf den Servicenamen gesetzt ist. Weitere Informationen zum Amazon EBS-Volume-Tagging finden Sie unter Tagging Amazon EBS-Volumes.

Die folgenden Optionen sind für dieses Feature erforderlich:

- Sie müssen sich für die neuen Formate Amazon Ressourcenname (ARN) und Ressourcen-Identifier (ID) anmelden (Opt-in). Weitere Informationen finden Sie unter Amazon Ressourcennamen (ARNs) und IDs.
- Wenn Sie die APIs verwenden, um einen Dienst zu erstellen oder eine Aufgabe auszuführen, müssen Sie `enableECSManagedTags` auf `true` für `run-task` und `create-service` setzen. Weitere Informationen finden Sie unter create-service und run-task unter AWS Command Line Interface -API-Referenz.
- Amazon ECS verwendet verwaltete Tags, um zu bestimmen, wann einige Features aktiviert sind, z. B. Cluster-Auto-Scaling. Wir empfehlen, Tags nicht manuell zu ändern, damit Amazon ECS die Funktionen effektiv verwalten kann.

Verwenden Sie Tags für die Abrechnung

AWS bietet ein Berichtstool namens Cost Explorer, mit dem Sie die Kosten und die Nutzung Ihrer Amazon ECS-Ressourcen analysieren können.

Verwenden Sie Cost Explorer, um Diagramme Ihrer Nutzung und Kosten anzuzeigen. Sie können die Daten der letzten 13 Monate anzeigen und prognostizieren, wie viel Sie wahrscheinlich für die nächsten drei Monate ausgeben werden. Sie können den Cost Explorer verwenden, um Muster zu sehen, wie viel Sie für AWS -Ressourcen im Zeitablauf ausgeben. Sie können es zum Beispiel verwenden, um Bereiche zu identifizieren, die eine genauere Untersuchung erfordern, sowie um Trends auszumachen, die Ihnen helfen, Ihre Kosten zu verstehen. Sie können auch Zeitbereiche für die Daten angeben und die Daten nach Tagen oder Monate anzeigen lassen.

Sie können von Amazon ECS verwaltete Tags oder von Benutzern hinzugefügte Tags für Ihren Kosten- und Nutzungsbericht verwenden. Weitere Informationen finden Sie unter Amazon-ECS-Nutzungsberichte.

Um die Kosten kombinierter Ressourcen anzuzeigen, können Sie Ihre Fakturierungsinformationen nach Ressourcen mit gleichen Tag (Markierung)-Schlüsselwerten strukturieren. Beispielsweise können Sie mehrere Ressourcen mit einem bestimmten Anwendungsnamen markieren und dann Ihre Fakturierungsinformationen so organisieren, dass Sie die Gesamtkosten dieser Anwendung über mehrere Services hinweg sehen können. Weitere Informationen zum Einrichten eines Kostenzuordnungsberichts mit Markierungen finden Sie unter Monatlicher Kostenzuordnungsbericht im Benutzerhandbuch für AWS Billing .

Außerdem können Sie die Option Split Cost Allocation Data aktivieren, um in Ihren Kosten- und Nutzungsberichten Daten zur CPU- und Speichernutzung auf Aufgabenebene zu erhalten. Weitere Informationen finden Sie unter Kosten- und Nutzungsberichte auf Aufgabenebene.

> **Note**
>
> Wenn Sie die Berichterstellung aktiviert haben, kann es bis zu 24 Stunden dauern, bis die Daten für den aktuellen Monat zur Ansicht verfügbar sind.

Hinzufügen von Tags zu Amazon ECS-Ressourcen

Sie können neue oder bestehende Aufgaben, Dienste, Aufgabendefinitionen oder Cluster taggen. Informationen zum Taggen Ihrer Container-Instances finden Sie unterHinzufügen von Tags zu einer Amazon ECS-Container-Instance.

> **Warning**
>
> Fügen Sie keine personenbezogenen Daten (Personally Identifiable Information, PII) oder andere vertrauliche Informationen in Tags hinzu. Tags sind für viele AWS Dienste zugänglich, auch für die Abrechnung. Tags sind nicht für private oder vertrauliche Daten gedacht.

Sie können die folgenden Ressourcen verwenden, um Tags anzugeben, wenn Sie die Ressource erstellen.

Aufgabe	Konsole	AWS CLI	API-Aktion
Führen Sie eine oder mehrere Aufgaben aus.	Eine Anwendung als Amazon ECS-Aufgabe ausführen	run-task	RunTask
Erstellen Sie einen Service.	Einen Amazon ECS-Service mithilfe der Konsole erstellen	create-service	CreateService
Einen Aufgabensatz erstellen.	Stellen Sie Amazon ECS-Services mithilfe eines Controllers eines Drittanbieters bereit	create-task-set	CreateTaskSet
Eine Aufgabendefinition registrieren.	the section called “Erstellen einer Aufgabendefinition mit der Konsole”	register-task-definition	RegisterTaskDefinition
Erstellen Sie einen Cluster.	Erstellen eines Amazon ECS-Clusters für den Starttyp Fargate	create-cluster	CreateCluster
Eine oder mehrere Container-Instances ausführen.	Starten einer Amazon ECS Linux-Container-Instance	run-instances	RunInstances

Hinzufügen von Tags zu vorhandenen Ressourcen (Amazon ECS-Konsole)

Sie können Tags, die Ihren Clustern, Services, Aufgaben und Aufgabendefinitionen zugeordnet sind, direkt von der Seite der Ressource aus hinzufügen oder löschen.

So ändern Sie ein Tag für eine einzelne Ressource

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie in der Navigationsleiste die aus, die Sie verwenden AWS-Region möchten.
3. Wählen Sie im Navigationsbereich einen Ressourcentyp aus (z. B. Clusters (Cluster)).
4. Wählen Sie die Ressource aus der Ressourcenliste aus, wählen Sie die Registerkarte Tags und dann Manage tags (Tags verwalten) aus.
5. Konfigurieren Sie Ihre Tags.

   [Ein Tag hinzufügen] Wählen Sie Add tag (Tag hinzufügen) und führen Sie dann das Folgende aus:

   - Geben Sie bei Key (Schlüssel) den Schlüsselnamen ein.
   - Geben Sie bei Value (Wert) den Wert des Schlüssels ein.
6. Wählen Sie Speichern.

Hinzufügen von Tags zu vorhandenen Ressourcen (AWS CLI)

Sie können ein oder mehrere Tags hinzufügen oder überschreiben, indem Sie die AWS CLI oder eine API verwenden.

- AWS CLI - Tag-Ressource
- API-Aktion - TagResource

Hinzufügen von Tags zu einer Amazon ECS-Container-Instance

Sie können mit einer der folgenden Methoden Tags zu Ihren Container-Instances zuweisen:

- Methode 1: Beim Erstellen der Container-Instance geben Sie mit der Amazon EC2-API, -CLI oder -Konsole die Tags an, indem Sie Benutzerdaten mit dem Konfigurationsparameter `ECS_CONTAINER_INSTANCE_TAGS` an die Instance übergeben. Dadurch werden Tags erstellt, die nur der Container-Instance in Amazon ECS zugeordnet sind. Sie können nicht mithilfe der Amazon

EC2-API angeboten werden. Weitere Informationen finden Sie unter Bootstrapping von Amazon ECS-Linux-Container-Instances zur Datenübergabe.

> **Important**
>
> Wenn Sie Ihre Container-Instances mit einer Amazon EC2 Auto Scaling-Gruppe starten, sollten Sie den Agent-Konfigurationsparameter ECS_CONTAINER_INSTANCE_TAGS verwenden, um Tags hinzuzufügen. Dies liegt an der Art und Weise, in der Amazon-EC2-Instances Tags hinzugefügt werden, die mit Auto-Scaling-Gruppen gestartet werden.

Nachfolgend ist ein Beispiel für ein Benutzerdatenskript angeführt, das Tags mit Ihrer Container-Instance verknüpft:

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Methode 2: Beim Erstellen Ihrer Container-Instance geben Sie mit der Amazon-EC2-API, -CLI oder -Konsole die Tags zunächst mit dem Parameter `TagSpecification.N` an. Übergeben Sie dann Benutzerdaten mithilfe des Container-Agent-Konfigurationsparameters `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` an die Instance. Dadurch werden sie von Amazon EC2 an Amazon ECS weitergegeben.

  Nachfolgend finden Sie ein Beispiel für ein Benutzerdaten-Skript, das die mit einer Amazon-EC2-Instance verknüpften Tags propagiert und die Instance mit einem Cluster namens `MyCluster` registriert:

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

  Um den Zugriff zu gewähren, damit Container-Instance-Tags von Amazon EC2 zu Amazon ECS propagieren werden können, fügen Sie manuell der IAM-Rolle der Amazon-ECS-Container-

Instance die folgenden Berechtigungen als Inline-Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- `ec2:DescribeTags`

Das folgende ist ein Beispiel einer Richtlinie mit der diese Berechtigungen hinzugefügt werden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Externe Container-Instances

Sie können mit einer der folgenden Methoden Tags zu Ihren externen Container-Instances zuweisen:

- Methode 1: Bevor Sie das Installationsskript ausführen, um Ihre externe Instance bei Ihrem Cluster zu registrieren, erstellen oder bearbeiten Sie die Konfigurationsdatei für den Amazon-ECS-Container Agent unter `/etc/ecs/ecs.config` und fügen Sie den `ECS_CONTAINER_INSTANCE_TAGS`-Konfigurationsparameter hinzu. Dadurch werden Tags erstellt, die der externen Instance zugeordnet sind.

  Es folgt ein Beispiel für die Syntax.

  ```
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  ```

- Methode 2 — Nachdem Ihre externe Instance in Ihrem Cluster registriert wurde, können Sie die verwenden, AWS Management Console um Tags hinzuzufügen. Weitere Informationen finden Sie unter Hinzufügen von Tags zu vorhandenen Ressourcen (Amazon ECS-Konsole).

Amazon-ECS-Nutzungsberichte

AWS bietet ein Berichtstool namens Cost Explorer, mit dem Sie die Kosten und die Nutzung Ihrer Amazon ECS-Ressourcen analysieren können.

Verwenden Sie Cost Explorer, um Diagramme Ihrer Nutzung und Kosten anzuzeigen. Sie können die Daten der letzten 13 Monate anzeigen und prognostizieren, wie viel Sie wahrscheinlich für die nächsten drei Monate ausgeben werden. Sie können den Cost Explorer verwenden, um Muster zu sehen, wie viel Sie für AWS -Ressourcen im Zeitablauf ausgeben. Sie können es zum Beispiel verwenden, um Bereiche zu identifizieren, die eine genauere Untersuchung erfordern, sowie um Trends auszumachen, die Ihnen helfen, Ihre Kosten zu verstehen. Sie können auch Zeitbereiche für die Daten angeben und die Daten nach Tagen oder Monate anzeigen lassen.

Die Messdaten in Ihrem Kosten- und Nutzungsbericht zeigen die Nutzung für alle Ihre Amazon-ECS-Aufgaben an. Die Messdaten umfassen die CPU-Auslastung als `vCPU-Hours` und die Speicherauslastung als `GB-Hours` für jede ausgeführte Aufgabe. Wie die Daten vorgelegt werden, hängt vom Starttyp der Aufgabe ab.

Bei Aufgaben, die den Fargate-Starttyp verwenden, zeigt die `lineItem/Operation`-Spalte `FargateTask` an und Sie sehen die mit jeder Aufgabe verbundenen Kosten.

Bei Aufgaben, die den EC2-Starttyp verwenden, zeigt die `lineItem/Operation`-Spalte `ECSTask-EC2` an und die Aufgaben sind mit keinen direkten Kosten verbunden. Die im Bericht angezeigten Messdaten, z. B. die Speichernutzung, stellen die Gesamtressourcen dar, die die Aufgabe in dem von Ihnen angegebenen Abrechnungszeitraum reserviert hat. Sie können diese Daten verwenden, um die Kosten Ihres zugrunde liegenden Clusters von Amazon-EC2-Instances zu ermitteln. Die Kosten- und Nutzungsdaten für Ihre Amazon-EC2-Instances werden separat unter dem Amazon-EC2-Service aufgeführt.

Sie können auch die von Amazon ECS verwalteten Tags verwenden, um den Service oder Cluster zu identifizieren, zu dem die einzelnen Aufgaben gehören. Weitere Informationen finden Sie unter Verwenden Sie Tags für die Abrechnung.

> **Important**
>
> Die Messdaten sind nur für Aufgaben sichtbar, die am oder nach dem 16. November 2018 gestartet wurden. Aufgaben, die vor diesem Datum gestartet wurden, zeigen keine Messdaten an.

Nachfolgend finden Sie ein Beispiel, wie Sie die Daten in einigen Feldern für die Kostenzuordnung mit Cost Explorer sortieren können:

- Clustername
- Service-Name
- Ressourcen-Tags
- Starttyp
- AWS-Region
- Verwendungstyp

Weitere Informationen zur Erstellung eines AWS Kosten- und Nutzungsberichts finden Sie unter AWS Kosten- und Nutzungsbericht im AWS Billing Benutzerhandbuch.

Kosten- und Nutzungsberichte auf Aufgabenebene

AWS Cost Management kann CPU- und Speichernutzungsdaten AWS Cost and Usage Report für jede Aufgabe auf Amazon ECS bereitstellen, einschließlich Aufgaben auf Fargate und Aufgaben auf EC2. Diese Daten werden als Split Cost Allocation Data bezeichnet. Sie können diese Daten verwenden, um Kosten und Nutzung für Anwendungen zu analysieren. Außerdem können Sie die Kosten mit Hilfe von Kostenzuordnungs-Tags und Kostenkategorien auf einzelne Geschäftseinheiten und Teams aufteilen und zuordnen. Weitere Informationen zu geteilten Kostenzuordnungsdaten finden Sie unter Grundlegendes zu geteilten Kostenzuordnungsdaten im AWS Cost and Usage Report Benutzerhandbuch.

Sie können sich für Split Cost Allocation Data auf Aufgabenebene für das Konto in AWS Cost Management Console entscheiden. Wenn Sie über ein Verwaltungskonto (Zahler) verfügen, können Sie sich vom Zahlerkonto aus dafür entscheiden, diese Konfiguration auf jedes verknüpfte Konto anzuwenden.

Nachdem Sie die Daten zur geteilten Kostenzuweisung eingerichtet haben, werden unter der splitLineItemÜberschrift des Berichts weitere Spalten angezeigt. Weitere Informationen finden Sie im AWS Cost and Usage Report Benutzerhandbuch unter Details zu Einzelposten aufteilen

Bei Aufgaben auf EC2 teilen diese Daten die Kosten der EC2-Instance auf der Grundlage der Ressourcennutzung oder -Reservierungen und der verbleibenden Ressourcen auf der Instance auf.

Die folgenden Voraussetzungen sind erforderlich:

- Stellen Sie den `ECS_DISABLE_METRICS` Amazon ECS-Agenten-Konfigurationsparameter auf ein`false`.

  Wenn diese Einstellung aktiviert ist`false`, sendet der Amazon ECS-Agent Metriken an Amazon CloudWatch. Unter Linux ist diese Einstellung `false` standardmäßig und Metriken werden an gesendet CloudWatch. Unter Windows ist diese Einstellung `true` standardmäßig, daher müssen Sie die Einstellung ändern, `false` um die Metriken CloudWatch zur Verwendung an AWS Cost Management zu senden. Weitere Informationen zur ECS-Agentenkonfiguration finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.
- Die Mindestversion von Docker für zuverlässige Metriken ist die Docker-Version v20.10.13 und neuer, die im Amazon-ECS-optimierten AMI 20220607 und neuer enthalten ist.

Um Split Cost Allocation Data verwenden zu können, müssen Sie einen Bericht erstellen und die Option Split Cost Allocation Data auswählen. Weitere Informationen finden Sie im Benutzerhandbuch unter Erstellen von Kosten- und Nutzungsberichten. AWS Cost and Usage Report

AWS Cost Management berechnet die Daten zur geteilten Kostenzuweisung anhand der CPU- und Speicherauslastung der Aufgabe. AWS Cost Management kann anstelle der Nutzung die CPU- und Speicherreservierung für die Aufgabe verwenden, wenn die Nutzung nicht verfügbar ist. Wenn Sie sehen, dass die CUR die Reservierungen verwendet, überprüfen Sie, ob Ihre Container-Instances die Voraussetzungen erfüllen und dass die Messwerte zur Nutzung der Task-Ressourcen unter angezeigt werden. CloudWatch

Überwachen von Amazon ECS

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon ECS und Ihren AWS Lösungen. Sie sollten Überwachungsdaten aus allen Teilen Ihrer AWS Lösung sammeln, damit Sie einen etwaigen Ausfall an mehreren Stellen leichter debuggen können. Bevor Sie mit der Überwachung von Amazon ECS beginnen, erstellen Sie einen Überwachungsplan, der Antworten auf die folgenden Fragen enthält:

- Was sind Ihre Ziele bei der Überwachung?
- Welche Ressourcen werden überwacht?
- Wie oft werden diese Ressourcen überwacht?
- Welche Überwachungstools werden verwendet?
- Wer soll die Überwachungsaufgaben ausführen?
- Wer soll benachrichtigt werden, wenn Fehler auftreten?

Die zur Verfügung gestellten Metriken hängen vom Starttyp der Aufgaben und Services in Ihren Clustern ab. Wenn Sie den Fargate-Starttyp für Ihre Services verwenden, werden Kennzahlen zur CPU- und Speicherauslastung bereitgestellt, um die Überwachung Ihrer Services zu unterstützen. Beim Amazon EC2-Starttyp müssen Sie Ihre eigenen EC2-Instances überwachen, die Ihre zugrunde liegende Infrastruktur bilden. Zusätzliche Kennzahlen zur Reservierung und Auslastung von CPU und Arbeitsspeicher werden im Cluster, Service und Task zur Verfügung gestellt.

Im nächsten Schritt legen Sie einen Ausgangswert für normale Amazon-ECS-Performance in Ihrer Umgebung fest, indem Sie die Leistung zu verschiedenen Zeiten und unter verschiedenen Lastbedingungen messen. Speichern Sie bei der Überwachung von Amazon ECS historische Überwachungsdaten, damit Sie diese mit aktuellen Leistungsdaten vergleichen, normale Leistungsmuster bestimmen, Leistungsprobleme erkennen und Methoden zur Fehlerbehebung ableiten können.

Zur Festlegung eines Grundwertes sollten Sie mindestens die folgenden Elemente überwachen:

- Die Reservierungs- und Auslastungsmetriken von CPU und Arbeitsspeicher für Ihre Amazon-ECS-Cluster
- Die Auslastungsmetriken von CPU und Arbeitsspeicher für Ihre Amazon-ECS-Services

  Weitere Informationen finden Sie unter Anzeigen von Amazon-ECS-Metriken.

Bewährte Methoden für die Überwachung von Amazon ECS

Verwenden Sie die folgenden bewährten Methoden für die Überwachung von Amazon ECS.

- Machen Sie die Überwachung zu einer Priorität, um kleine Probleme zu vermeiden, bevor sie zu großen werden
- Erstellen Sie einen Überwachungsplan, der Antworten auf die folgende Frage enthält
  - Was sind Ihre Ziele bei der Überwachung?
  - Welche Ressourcen werden überwacht?
  - Wie oft werden diese Ressourcen überwacht?
  - Welche Überwachungstools werden verwendet?
  - Wer soll die Überwachungsaufgaben ausführen?
  - Wer soll benachrichtigt werden, wenn Fehler auftreten?
- Automatisieren Sie die Überwachung so weit wie möglich.
- Überprüfen Sie die Amazon ECS-Protokolldateien. Weitere Informationen finden Sie unter Amazon ECS-Container-Agent-Protokolle anzeigen.

Überwachungstools für Amazon ECS

AWS bietet verschiedene Tools, mit denen Sie Amazon ECS überwachen können. Sie können einige dieser Tools so konfigurieren, dass diese die Überwachung für Sie übernehmen, während bei anderen Tools ein manuelles Eingreifen nötig ist. Wir empfehlen, dass Sie die Überwachungsaufgaben möglichst automatisieren.

Automatisierte Überwachungstools

Sie können die folgenden automatisierten Tools zur Überwachung von Amazon ECS verwenden und auftretende Probleme melden:

- CloudWatch Amazon-Alarme — Überwachen Sie eine einzelne Metrik über einen von Ihnen angegebenen Zeitraum und führen Sie eine oder mehrere Aktionen aus, die auf dem Wert der Metrik im Verhältnis zu einem bestimmten Schwellenwert über mehrere Zeiträume basieren. Die Aktion ist eine Benachrichtigung, die an ein Amazon Simple Notification Service (Amazon SNS) -Thema oder eine Amazon EC2 Auto Scaling Scaling-Richtlinie gesendet wird. CloudWatch Alarme lösen keine Aktionen aus, nur weil sie sich in einem bestimmten Status befinden. Der Status muss

sich geändert haben und für eine bestimmte Anzahl von Zeiträumen beibehalten worden sein. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS mit CloudWatch .

Für Dienste mit Aufgaben, die den Starttyp Fargate verwenden, können Sie CloudWatch Alarme verwenden, um die Aufgaben in Ihrem Service auf der Grundlage von CloudWatch Metriken wie CPU- und Speicherauslastung zu vergrößern und zu skalieren. Weitere Informationen finden Sie unter Skalieren Sie Ihren Amazon ECS-Service automatisch.

Bei Clustern mit Aufgaben oder Diensten, die den EC2-Starttyp verwenden, können Sie CloudWatch Alarme verwenden, um die Container-Instances auf der Grundlage von CloudWatch Metriken wie der Cluster-Speicherreservierung zu vergrößern und zu skalieren.

Für Ihre Container-Instances, die mit dem Amazon ECS-optimierten Amazon Linux AMI gestartet wurden, können Sie CloudWatch Logs verwenden, um verschiedene Logs aus Ihren Container-Instances bequem an einem zentralen Ort anzuzeigen. Sie müssen den CloudWatch Agenten auf Ihren Container-Instances installieren. Weitere Informationen finden Sie unter Herunterladen und Konfigurieren des CloudWatch Agenten über die Befehlszeile im CloudWatch Amazon-Benutzerhandbuch. Außerdem müssen Sie der `ecsInstanceRole`-Rolle die Richtlinie `ECS-CloudWatchLogs` hinzufügen. Weitere Informationen finden Sie unter Überwachung der Berechtigungen für Container-Instances.

- Amazon CloudWatch Logs — Überwachen Sie die Protokolldateien aus den Containern in Ihren Amazon ECS-Aufgaben, speichern Sie sie und greifen Sie darauf zu, indem Sie den `awslogs` Protokolltreiber in Ihren Aufgabendefinitionen angeben. Weitere Informationen finden Sie unter Amazon ECS-Protokolle senden an CloudWatch .

  Sie können von Ihren Amazon-ECS-Container-Instances aus die Betriebssystem- und Amazon-ECS-Container-Agent-Protokolldateien überwachen, speichern und darauf zugreifen. Diese Methode für den Zugriff auf Protokolle kann für Container verwendet werden, die den EC2-Starttyp verwenden.

- Amazon CloudWatch Events — Ordnen Sie Ereignisse zu und leiten Sie sie an eine oder mehrere Zielfunktionen oder Streams weiter, um Änderungen vorzunehmen, Statusinformationen zu erfassen und Korrekturmaßnahmen zu ergreifen. Weitere Informationen finden Sie Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge in diesem Leitfaden und unter Was ist Amazon CloudWatch Events? im Amazon CloudWatch Events-Benutzerhandbuch.

- Container Insights — Sammeln, aggregieren und fassen Sie Metriken und Logs aus Ihren containerisierten Anwendungen und Microservices zusammen. Container Insights sammelt Daten als Leistungsprotokollereignisse unter Verwendung des eingebetteten Metrikformats. Bei

diesen Leistungsprotokollereignissen handelt es sich um Einträge, die ein strukturiertes JSON-Schema verwenden, mit dem Daten mit hoher Kardinalität in großem Umfang aufgenommen und gespeichert werden können. Aus diesen Daten werden aggregierte Metriken auf Cluster-, Aufgaben- und Serviceebene als Metriken CloudWatch erstellt. CloudWatch Die von Container Insights gesammelten Metriken sind in CloudWatch automatischen Dashboards verfügbar und können auch im Bereich Metriken der Konsole eingesehen werden. CloudWatch

- AWS CloudTrail Protokollüberwachung — Teilen Sie Protokolldateien zwischen Konten, überwachen CloudTrail Sie Protokolldateien in Echtzeit, indem Sie sie an CloudWatch Logs senden, schreiben Sie Protokollverarbeitungsanwendungen in Java und stellen Sie sicher, dass sich Ihre Protokolldateien nach der Lieferung von CloudTrail nicht geändert haben. Weitere Informationen finden Sie Amazon ECS-API-Aufrufe protokollieren mit AWS CloudTrail in diesem Handbuch und Arbeiten mit CloudTrail Protokolldateien im AWS CloudTrail Benutzerhandbuch.
- Runtime Monitoring — Erkennen Sie Bedrohungen für Cluster und Container in Ihrer AWS Umgebung. Runtime Monitoring verwendet einen GuardDuty Sicherheitsagenten, der die Laufzeit einzelner Amazon ECS-Workloads transparent macht, z. B. Dateizugriff, Prozessausführung und Netzwerkverbindungen.

Manuelle Überwachungstools

Ein weiterer wichtiger Teil der Überwachung von Amazon ECS ist die manuelle Überwachung der Elemente, die von den CloudWatch Alarmen nicht abgedeckt werden. Die Dashboards CloudWatch Trusted Advisor, und andere AWS Konsolen-Dashboards bieten einen at-a-glance Überblick über den Zustand Ihrer AWS Umgebung. Wir empfehlen, dass Sie auch die Protokolldateien auf Ihren Container-Instances und die Container in Ihren Aufgaben überprüfen.

- Amazon ECS-Konsole:
  - Cluster-Metriken für den EC2-Starttyp
  - Servicemetriken
  - Servicestatus
  - Ereignisse bei der Bereitstellung von Diensten
- CloudWatch Startseite:
  - Aktuelle Alarme und Status
  - Diagramme mit Alarmen und Ressourcen
  - Servicestatus

Darüber hinaus können CloudWatch Sie Folgendes verwenden:

- Erstellen von benutzerdefinierten Dashboards zur Überwachung des gewünschten Services.
- Aufzeichnen von Metrikdaten, um Probleme zu beheben und Trends zu erkennen.
- Suchen und durchsuchen Sie alle Ihre AWS Ressourcenmetriken.
- Erstellen und Bearbeiten von Alarmen, um über Probleme benachrichtigt zu werden.

- Container-Integritätsprüfung — Dies sind Befehle, die lokal in einem Container ausgeführt werden und den Zustand und die Verfügbarkeit von Anwendungen überprüfen. Sie konfigurieren diese pro Container in Ihrer Aufgabendefinition.
- AWS Trusted Advisor kann Ihnen helfen, Ihre AWS Ressourcen zu überwachen, um Leistung, Zuverlässigkeit, Sicherheit und Wirtschaftlichkeit zu verbessern. Vier Trusted Advisor Checks stehen allen Benutzern zur Verfügung; mehr als 50 Checks stehen Benutzern mit einem Business- oder Enterprise-Supportplan zur Verfügung. Weitere Informationen finden Sie unter AWS Trusted Advisor.

  Trusted Advisor hat diese Prüfungen, die sich auf Amazon ECS beziehen:

  - Eine Fehlertoleranz, die darauf hinweist, dass ein Service in einer einzigen Availability Zone ausgeführt wird.
  - Eine Fehlertoleranz, die darauf hinweist, dass Sie die Spread Placement-Strategie nicht für mehrere Availability Zones verwendet haben.
- AWS Compute Optimizer ist ein Service, der die Konfiguration und Nutzungskennzahlen Ihrer AWS Ressourcen analysiert. Es berichtet, ob Ihre Ressourcen optimal sind und generiert Optimierungsempfehlungen, um die Kosten zu senken und die Leistung Ihrer Workloads zu verbessern.

  Weitere Informationen finden Sie unter AWS Compute Optimizer Empfehlungen für Amazon ECS.

Überwachen Sie Amazon ECS mit CloudWatch

Sie können Ihre Amazon ECS-Ressourcen mithilfe von Amazon überwachen CloudWatch, das Rohdaten von Amazon ECS sammelt und zu lesbaren Metriken nahezu in Echtzeit verarbeitet. Diese Statistiken werden für einen Zeitraum von zwei Wochen aufgezeichnet, damit Sie auf Verlaufsinformationen zugreifen können und einen besseren Überblick darüber erhalten, wie Ihre Clusters oder Services ausgeführt werden. Amazon ECS-Metrikdaten werden automatisch

CloudWatch in Zeitabständen von 1 Minute an gesendet. Weitere Informationen zu CloudWatch finden Sie im CloudWatch Amazon-Benutzerhandbuch.

Amazon ECS bietet kostenlose Metriken für Cluster und Services. Gegen eine zusätzliche Gebühr können Sie Amazon ECS CloudWatch Container Insights für Ihren Cluster für aufgabenbezogene Metriken, einschließlich CPU-, Arbeitsspeicher- und EBS-Dateisystemauslastung, aktivieren. Weitere Informationen zu Container Insights finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.

Überlegungen

Folgendes sollte bei der Verwendung von Amazon CloudWatch ECS-Metriken berücksichtigt werden.

- Jeder auf Fargate gehostete Amazon ECS-Service verfügt automatisch über Kennzahlen zur CloudWatch CPU- und Speicherauslastung, sodass Sie keine manuellen Schritte unternehmen müssen.

- Für alle Amazon ECS-Aufgaben oder -Services, die auf Amazon EC2 EC2-Instances gehostet werden, benötigt die Amazon EC2 EC2-Instance Version `1.4.0` oder höher (Linux) `1.0.0` oder höher (Windows) des Container-Agenten, damit CloudWatch Metriken generiert werden können. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.
- Die Mindestversion von Docker für zuverlässige CloudWatch Metriken ist die Docker-Version und neuer. `20.10.13`
- Ihre Amazon EC2 EC2-Instances benötigen außerdem die `ecs:StartTelemetrySession` Genehmigung für die IAM-Rolle, mit der Sie Ihre Amazon EC2 EC2-Instances starten. Wenn Sie Ihre Amazon ECS-Container-Instance-IAM-Rolle erstellt haben, bevor CloudWatch Metriken für Amazon ECS verfügbar waren, müssen Sie diese Berechtigung möglicherweise hinzufügen. Informationen zur IAM-Rolle der Container-Instance und zum Anhängen der verwalteten IAM-Richtlinie für Container-Instances finden Sie unter. IAM-Rolle für Amazon-ECS-Container-Instance
- Sie können die Erfassung von CloudWatch Metriken auf Ihren Amazon EC2 EC2-Instances deaktivieren, indem Sie dies `ECS_DISABLE_METRICS=true` in Ihrer Amazon ECS-Container-Agent-Konfiguration festlegen. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Empfohlene Metriken

Amazon ECS bietet kostenlose CloudWatch Metriken, mit denen Sie Ihre Ressourcen überwachen können. Die CPU- und Speicherreservierung und die CPU-, Arbeitsspeicher- und EBS-Dateisystemauslastung in Ihrem gesamten Cluster sowie die CPU-, Speicher- und EBS-Dateisystemauslastung der Services in Ihren Clustern können anhand dieser Metriken gemessen werden. Für Ihre GPU-Workloads können Sie Ihre GPU-Reservierung in Ihrem Cluster messen.

Die Infrastruktur, auf der Ihre Amazon ECS-Aufgaben in Ihren Clustern gehostet werden, bestimmt, welche Metriken verfügbar sind. Für Aufgaben, die auf der Fargate-Infrastruktur gehostet werden, stellt Amazon ECS Kennzahlen zur CPU-, Arbeitsspeicher- und EBS-Dateisystemauslastung bereit, die Sie bei der Überwachung Ihrer Services unterstützen. Für Aufgaben, die auf EC2-Instances gehostet werden, bietet Amazon ECS CPU-, Arbeitsspeicher- und GPU-Reservierungsmetriken sowie Metriken zur CPU- und Speicherauslastung auf Cluster- und Serviceebene. Die Amazon EC2-Instances, die Ihre zugrunde liegende Infrastruktur bilden, müssen Sie separat überwachen. Weitere Informationen zur Überwachung Ihrer Amazon EC2-Instances finden Sie unter Überwachung von Amazon EC2 im Amazon EC2 EC2-Benutzerhandbuch.

Informationen zu den empfohlenen Alarmen für die Verwendung mit Amazon ECS finden Sie in einem der folgenden Abschnitte im Amazon CloudWatch Logs-Benutzerhandbuch:

- Amazon ECS
- Amazon ECS mit Container Insights

Anzeigen von Amazon-ECS-Metriken

Nachdem Sie Ressourcen in Ihrem Cluster ausgeführt haben, können Sie die Metriken auf dem Amazon ECS und den CloudWatch Konsolen anzeigen. Die Amazon-ECS-Konsole bietet eine 24-Stunden-Maximal-, Minimal- und Durchschnittsansicht Ihrer Cluster und Service-Metriken. Die CloudWatch Konsole bietet eine detaillierte und anpassbare Anzeige Ihrer Ressourcen sowie der Anzahl der laufenden Aufgaben in einem Service.

Amazon-ECS-Konsole

Die CPU- und Speichernutzungsmetriken des Amazon-ECS-Service sind in der Amazon-ECS-Konsole verfügbar. Die Anzeige für die Service-Metriken stellt die Durchschnitts-, Mindest- und Höchstwerte der letzten 24 Stunden dar, wobei Datenpunkte in 5-Minuten-Intervallen verfügbar sind. Weitere Informationen finden Sie unter Kennzahlen zur Nutzung des Amazon ECS-Service.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie den Cluster aus, für die Sie Metriken anzeigen möchten.
3. Legen Sie fest, welche Metriken angezeigt werden sollen.

Um Metriken anzuzeigen von	Schritte
Cluster	Wählen Sie auf der Seite mit den Cluster-Details die Registerkarte Metriken aus. Es gibt auch einen Link zur CloudWatch Konsole, über den Sie Ihre CloudWatch Container Insights-Metriken einsehen können, sofern Sie diese aktiviert haben.
Services	Wählen Sie auf der Seite mit den Cluster-Details auf der Registerkarte Dienste den Service aus. Die Metriken sind dann auf der Registerk arte Health und Metriken verfügbar.

CloudWatch Konsole

Für den Starttyp Fargate können die Amazon ECS-Servicemetriken auch auf der CloudWatch Konsole angezeigt werden. Die Konsole bietet die detaillierteste Ansicht von Amazon-ECS-Metriken, und Sie können die Anzeige an Ihre Bedürfnisse anpassen. Sie können die Service-Auslastung und die Anzahl der RUNNING-Aufgaben des Service anzeigen.

Für den EC2-Starttyp können Amazon ECS-Cluster- und Service-Metriken auch auf der CloudWatch Konsole angezeigt werden. Die Konsole bietet die detaillierteste Ansicht von Amazon-ECS-Metriken, und Sie können die Anzeige an Ihre Bedürfnisse anpassen.

Informationen zum Anzeigen der Metriken finden Sie im CloudWatch Amazon-Benutzerhandbuch unter Verfügbare Metriken anzeigen.

Amazon CloudWatch ECS-Metriken

Sie können CloudWatch Nutzungsmetriken verwenden, um einen Überblick über die Ressourcennutzung Ihres Kontos zu erhalten. Verwenden Sie diese Kennzahlen, um Ihre aktuelle Servicenutzung in CloudWatch Diagrammen und Dashboards zu visualisieren.

`CPUReservation`

Der Prozentsatz der CPU-Einheiten, die im Cluster oder Dienst reserviert sind.

Die CPU-Reservierung (gefiltert nach`ClusterName`) wird als die Gesamtzahl der CPU-Einheiten gemessen, die durch Amazon ECS-Aufgaben auf dem Cluster reserviert sind, geteilt durch die Gesamtzahl der CPU-Einheiten für alle im Cluster registrierten Amazon EC2 EC2-Instances. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die CPU-Reservierungsmetriken aus. Die Metrik wird nur für Aufgaben unterstützt, die auf einer Amazon EC2 EC2-Instance gehostet werden.

Gültige Dimensionen: `ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Einheit: Prozent.

`CPUUtilization`

Der Prozentsatz der CPU-Einheiten, der vom Cluster oder Dienst verwendet wird.

Die CPU-Auslastung auf Clusterebene (gefiltert nach`ClusterName`) wird als die Gesamtzahl der CPU-Einheiten gemessen, die von Amazon ECS-Aufgaben auf dem Cluster verwendet werden, geteilt durch die Gesamtzahl der CPU-Einheiten für alle im Cluster registrierten Amazon EC2 EC2-Instances. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die CPU-Reservierungsmetriken aus. Die Metrik auf Clusterebene wird nur für Aufgaben unterstützt, die auf einer Amazon EC2 EC2-Instance gehostet werden.

Die CPU-Auslastung auf Service-Ebene (gefiltert nach`ClusterName`,`ServiceName`) wird als die Gesamtzahl der CPU-Einheiten gemessen, die von den Aufgaben verwendet werden, die zum Service gehören, geteilt durch die Gesamtzahl der CPU-Einheiten, die für die Aufgaben reserviert sind, die zum Service gehören. Die Service-Level-Metrik wird für Aufgaben unterstützt, die auf Amazon EC2 EC2-Instances und Fargate gehostet werden.

Gültige Dimensionen: `ClusterName`, `ServiceName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Einheit: Prozent.

`MemoryReservation`

Der Prozentsatz des Arbeitsspeichers, der von laufenden Tasks im Cluster reserviert ist.

Die Cluster-Speicherreservierung wird als Gesamtspeicher gemessen, der für Amazon ECS-Aufgaben auf dem Cluster reserviert ist, geteilt durch die Gesamtspeichermenge für alle im Cluster registrierten Amazon EC2 EC2-Instances. Diese Metrik kann nur nach gefiltert `ClusterName` werden. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die Speicherreservierungsmetriken aus. Die Metrik zur Speicherreservierung auf Clusterebene wird nur für Aufgaben unterstützt, die auf einer Amazon EC2 EC2-Instance gehostet werden.

> **Note**
>
> Wenn bei der Berechnung der Speicherauslastung angegeben `MemoryReservation` wird, wird sie anstelle des Gesamtspeichers in der Berechnung verwendet.

Gültige Dimensionen: `ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Einheit: Prozent.

`MemoryUtilization`

Der Prozentsatz des Speichers, der vom Cluster oder Dienst genutzt wird.

Die Speicherauslastung auf Clusterebene (gefiltert nach`ClusterName`) wird als Gesamtspeicher gemessen, der von Amazon ECS-Aufgaben auf dem Cluster verwendet wird, geteilt durch den Gesamtspeicher für alle im Cluster registrierten Amazon EC2 EC2-Instances. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die Kennzahlen zur Speichernutzung aus. Die Metrik auf Clusterebene wird nur für Aufgaben unterstützt, die auf einer Amazon EC2 EC2-Instance gehostet werden.

Die Speicherauslastung auf Service-Ebene (gefiltert nach`ClusterName`,`ServiceName`) wird als der gesamte Arbeitsspeicher gemessen, der von den Aufgaben verwendet wird, die zum

Service gehören, geteilt durch den gesamten Speicher, der für die Aufgaben reserviert ist, die zum Service gehören. Die Service-Level-Metrik wird für Aufgaben unterstützt, die auf Amazon EC2 EC2-Instances und Fargate gehostet werden.

Gültige Dimensionen: `ClusterName`, `ServiceName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Einheit: Prozent.

`EBSFilesystemUtilization`

Der Prozentsatz des Amazon EBS-Dateisystems, der von Aufgaben in einem Service verwendet wird.

Die Service-Level-Metrik zur Nutzung des EBS-Dateisystems (gefiltert nach`ClusterName`,`ServiceName`) wird als Gesamtmenge des EBS-Dateisystems gemessen, das von den Aufgaben verwendet wird, die zum Service gehören, geteilt durch die Gesamtmenge an EBS-Dateisystemspeicher, die allen Aufgaben zugewiesen ist, die zum Service gehören. Die Service-Level-Metrik zur Nutzung des EBS-Dateisystems ist nur für Aufgaben verfügbar, die auf Amazon EC2 EC2-Instances (unter Verwendung der Container-Agent-Version`1.79.0`) und Fargate (unter Verwendung der Plattformversion`1.4.0`) gehostet werden und an die ein EBS-Volume angehängt ist.

> **Note**
>
> Für Aufgaben, die auf Fargate gehostet werden, gibt es Speicherplatz auf der Festplatte, der nur von Fargate verwendet wird. Für den Speicherplatz, den Fargate verwendet, fallen keine Kosten an, aber Sie werden diesen zusätzlichen Speicherplatz mithilfe von Tools wie `df` sehen.

Gültige Dimensionen: `ClusterName, ServiceName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Einheit: Prozent.

`GPUReservation`

Der Prozentsatz der insgesamt verfügbaren GPUs, die von laufenden Aufgaben im Cluster reserviert sind.

Die GPU-Reservierungsmetrik auf Clusterebene wird als die Anzahl der GPUs gemessen, die durch Amazon ECS-Aufgaben auf dem Cluster reserviert wurden, geteilt durch die Gesamtzahl der GPUs, die auf allen Amazon EC2 EC2-Instances mit im Cluster registrierten GPUs verfügbar waren. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die GPU-Reservierungsmetriken aus.

Gültige Dimensionen: `ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum

Alle Statistiken: Durchschnitt, Minimum, Maximum, Summe, Stichprobenanzahl.

Einheit: Prozent.

`ActiveConnectionCount`

Die Gesamtzahl der gleichzeitig aktiven Verbindungen von Clients zu den Service-Connect-Proxys für Amazon ECS, die in Aufgaben ausgeführt werden, die den ausgewählten `DiscoveryName` gemeinsam nutzen.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `DiscoveryName` und `DiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`NewConnectionCount`

Die Gesamtzahl der gleichzeitig aktiven Verbindungen von Clients zu den Service-Connect-Proxys für Amazon ECS, die in Aufgaben ausgeführt werden, die den ausgewählten `DiscoveryName` gemeinsam nutzen.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `DiscoveryName` und `DiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`ProcessedBytes`

Die Gesamtzahl der Bytes des eingehenden Datenverkehrs, die von den Service-Connect-Proxys verarbeitet werden.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `DiscoveryName` und `DiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Byte.

`RequestCount`

Die Anzahl der eingehenden Datenverkehrsanfragen, die von den Service-Connect-Proxys verarbeitet wurden.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Sie müssen auch die Port-Zuordnung in Ihrer Aufgabendefinition konfigurieren`appProtocol`.

Gültige Dimensionen: `DiscoveryName` und `DiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`GrpcRequestCount`

Die Anzahl von gRPC-Anfragen für eingehenden Datenverkehr, die von den Service-Connect-Proxys verarbeitet werden.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben und das `appProtocol` in der Portzuordnung in der Aufgabendefinition `GRPC` lautet.

Gültige Dimensionen: `DiscoveryName` und `DiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`HTTPCode_Target_2XX_Count`

Die Anzahl der HTTP-Antwortcodes mit den Nummern 200 bis 299, die von den Anwendungen in diesen Aufgaben generiert wurden. Diese Aufgaben sind die Ziele. Diese Metrik zählt nur die Antworten, die von den Anwendungen in diesen Aufgaben an die Service-Connect-Proxys gesendet wurden, nicht die direkt gesendeten Antworten.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben und das `appProtocol` in der Portzuordnung in der Aufgabendefinition `HTTP` oder `HTTP2` lautet.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`HTTPCode_Target_3XX_Count`

Die Anzahl der HTTP-Antwortcodes mit den Nummern 300 bis 399, die von den Anwendungen in diesen Aufgaben generiert wurden. Diese Aufgaben sind die Ziele. Diese Metrik zählt nur die Antworten, die von den Anwendungen in diesen Aufgaben an die Service-Connect-Proxys gesendet wurden, nicht die direkt gesendeten Antworten.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben und das `appProtocol` in der Portzuordnung in der Aufgabendefinition `HTTP` oder `HTTP2` lautet.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`HTTPCode_Target_4XX_Count`

Die Anzahl der HTTP-Antwortcodes mit den Nummern 400 bis 499, die von den Anwendungen in diesen Aufgaben generiert wurden. Diese Aufgaben sind die Ziele. Diese Metrik zählt nur die Antworten, die von den Anwendungen in diesen Aufgaben an die Service-Connect-Proxys gesendet wurden, nicht die direkt gesendeten Antworten.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben und das `appProtocol` in der Portzuordnung in der Aufgabendefinition `HTTP` oder `HTTP2` lautet.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe

Einheit: Anzahl.

`HTTPCode_Target_5XX_Count`

Die Anzahl der HTTP-Antwortcodes mit den Nummern 500 bis 599, die von den Anwendungen in diesen Aufgaben generiert wurden. Diese Aufgaben sind die Ziele. Diese Metrik zählt nur die Antworten, die von den Anwendungen in diesen Aufgaben an die Service-Connect-Proxys gesendet wurden, nicht die direkt gesendeten Antworten.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben und das `appProtocol` in der Portzuordnung in der Aufgabendefinition `HTTP` oder `HTTP2` lautet.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`RequestCountPerTarget`

Die durchschnittliche Anzahl von Anfragen, die von jedem Ziel empfangen werden, das den ausgewählten `DiscoveryName` gemeinsam hat.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt.

Einheit: Anzahl.

`TargetProcessedBytes`

Die Gesamtzahl der von den Service-Connect-Proxys verarbeiteten Bytes.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Byte.

`TargetResponseTime`

Die Latenz der Verarbeitung der Anwendungsanfrage. Die verstrichene Zeit in Millisekunden, nachdem die Anfrage den Service-Connect-Proxy in der Zielaufgabe erreicht hat, bis eine Antwort von der Zielanwendung zurück an den Proxy empfangen wird.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `TargetDiscoveryName` und `TargetDiscoveryName, ServiceName, ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum.

Alle Statistiken: Durchschnitt, Minimum, Maximum, Summe, Stichprobenanzahl.

Einheit: Millisekunden.

`ClientTLSNegotiationErrorCount`

Gibt an, wie oft die TLS-Verbindung insgesamt fehlgeschlagen ist. Diese Metrik wird nur verwendet, wenn TLS aktiviert ist.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Abmessungen: `DiscoveryName` und`DiscoveryName`,`ServiceName`,`ClusterName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

`TargetTLSNegotiationErrorCount`

Die Gesamtzahl der Fehlschläge der TLS-Verbindung aufgrund fehlender Client-Zertifikate, fehlgeschlagener AWS Private CA Überprüfungen oder fehlgeschlagener SAN-Überprüfungen. Diese Metrik wird nur verwendet, wenn TLS aktiviert ist.

Diese Metrik ist nur verfügbar, wenn Sie Amazon ECS Service Connect konfiguriert haben.

Gültige Dimensionen: `ServiceNameClusterName`, `TargetDiscoveryName` und`TargetDiscoveryName`.

Nützliche Statistiken: Durchschnitt, Minimum, Maximum, Summe.

Einheit: Anzahl.

Dimensionen für Amazon-ECS-Metriken

Amazon-ECS-Metriken verwenden den `AWS/ECS`-Namespace und stellen Metriken für folgende Dimension(en) bereit. Amazon ECS sendet nur Metriken für Ressourcen, die Aufgaben im Status `RUNNING` haben. Wenn Sie beispielsweise einen Cluster mit einem Dienst haben, dieser Dienst jedoch keine Aufgaben in einem bestimmten `RUNNING` Status hat, werden keine Metriken gesendet CloudWatch. Wenn Sie über zwei Services verfügen, und einer von diesen hat ausgeführte Aufgaben und der andere nicht, werden nur die Metriken für den Service mit ausgeführten Aufgaben gesendet.

`ClusterName`

Diese Dimension filtert die angeforderten Daten für alle Ressourcen in einem angegebenen Cluster. Alle Amazon-ECS-Metriken werden von `ClusterName` gefiltert.

`ServiceName`

Diese Dimension filtert die angeforderten Daten für alle Ressourcen in einem angegebenen Service innerhalb eines angegebenen Clusters.

`DiscoveryName`

Diese Dimension filtert die Daten, die Sie für Datenverkehrsmetriken zu einem bestimmten Service-Connect-Erkennungsnamen anfragen, über alle Amazon-ECS-Cluster hinweg.

Berücksichtigen Sie, dass ein bestimmter Port in einem ausgeführten Container mehrere Erkennungsnamen haben kann.

`DiscoveryName, ServiceName, ClusterName`

Diese Dimension filtert die Daten, die Sie für Datenverkehrsmetriken zu einem bestimmten Erkennungsnamen von Service Connect anfragen, über Aufgaben, die diesen Erkennungsnamen haben und die von diesem Service in diesem Cluster erstellt werden, hinweg.

Verwenden Sie diese Dimension, um die Metriken zum eingehenden Datenverkehr für einen bestimmten Service anzuzeigen, wenn Sie denselben Erkennungsnamen in mehreren Services in verschiedenen Namespaces wiederverwendet haben.

Berücksichtigen Sie, dass ein bestimmter Port in einem ausgeführten Container mehrere Erkennungsnamen haben kann.

`TargetDiscoveryName`

Diese Dimension filtert die Daten, die Sie für Datenverkehrsmetriken zu einem bestimmten Service-Connect-Erkennungsnamen anfragen, über alle Amazon-ECS-Cluster hinweg.

Im Gegensatz zu `DiscoveryName` messen diese Datenverkehrsmetriken nur den eingehenden Datenverkehr zu diesem `DiscoveryName`, der von anderen Amazon-ECS-Aufgaben stammt, die über eine Service-Connect-Konfiguration in diesem Namespace verfügen. Dazu gehören Aufgaben, die von Services mit einer reinen Client- oder Client-Server-Konfiguration von Service Connect durchgeführt werden.

Berücksichtigen Sie, dass ein bestimmter Port in einem ausgeführten Container mehrere Erkennungsnamen haben kann.

`TargetDiscoveryName, ServiceName, ClusterName`

Diese Dimension filtert die Daten, die Sie für Datenverkehrsmetriken anfragen, auf einen bestimmten Erkennungsnamen von Service Connect, zählt aber nur Datenverkehr von Aufgaben, die von diesem Service in diesem Cluster erstellt wurden.

Verwenden Sie diese Dimension, um die Metriken zum eingehenden Datenverkehr anzuzeigen, die von einem bestimmten Client in einem anderen Service stammen.

Im Gegensatz zu `DiscoveryName, ServiceName, ClusterName` messen diese Datenverkehrsmetriken nur den eingehenden Datenverkehr zu diesem `DiscoveryName`, der von anderen Amazon-ECS-Aufgaben stammt, die über eine Service-Connect-Konfiguration in diesem Namespace verfügen. Dazu gehören Aufgaben, die von Services mit einer reinen Client- oder Client-Server-Konfiguration von Service Connect durchgeführt werden.

Berücksichtigen Sie, dass ein bestimmter Port in einem ausgeführten Container mehrere Erkennungsnamen haben kann.

AWS Fargate Nutzungsmetriken

Sie können CloudWatch Nutzungsmetriken verwenden, um einen Überblick über die Ressourcennutzung Ihres Kontos zu erhalten. Verwenden Sie diese Kennzahlen, um Ihre aktuelle Servicenutzung in CloudWatch Diagrammen und Dashboards zu visualisieren.

AWS Fargate Nutzungsmetriken entsprechen den AWS Servicekontingenten. Sie können Alarme konfigurieren, mit denen Sie benachrichtigt werden, wenn sich Ihre Nutzung einem Servicekontingent nähert. Weitere Hinweise zu Servicekontingenten für Fargate finden Sie unter AWS Fargate Servicekontingenten.

AWS Fargate veröffentlicht die folgenden Metriken im `AWS/Usage` Namespace.

Metrik	Beschreibung
`ResourceCount`	Die Gesamtanzahl der angegebenen Ressourcen, die in Ihrem Konto ausgeführt werden. Die Ressource wird durch die Dimensionen definiert, die der Metrik zugeordnet sind.

Die folgenden Dimensionen werden verwendet, um die Nutzungsmetriken zu verfeinern, die von AWS Fargate veröffentlicht werden.

Dimension	Beschreibung
`Service`	Der Name des AWS Dienstes, der die Ressource enthält. Für AWS Fargate -Nutzungsmetriken lautet der Wert für diese Dimension `Fargate`.
`Type`	Der Typ von Entität, die gemeldet wird. Derzeit ist der einzig gültige Wert für AWS Fargate NutzungsmetrikenResource.
`Resource`	Der Typ der Ressource, die ausgeführt wird. Der Typ der Ressource, die ausgeführt wird. Derzeit ist der einzig gültige Wert für AWS Fargate Nutzungsmetriken der WertvCPU, der Informationen über die laufenden Instances zurückgibt.
`Class`	Die Klasse der nachverfolgten Ressource. Die Klasse der nachverfolgten Ressource. Für AWS Fargate Nutzungsmetriken mit vCPU als Wert der Resource-Dimension sind die gültigen Werte `Standard/OnDemand` und`Standard/Spot` .

Sie können die Service-Kontingents-Konsole verwenden, um Ihre Nutzung in einem Diagramm zu visualisieren und Alarme zu konfigurieren, die Sie benachrichtigen, wenn sich Ihre AWS Fargate

Nutzung einem Servicekontingent nähert. Informationen zum Erstellen eines CloudWatch Alarms, der Sie benachrichtigt, wenn Sie sich einem Kontingentschwellenwert nähern, finden Sie unter Servicequotas und CloudWatch Amazon-Alarme im Service Quotas Benutzerhandbuch

.

Amazon ECS-Cluster-Reservierungsmetriken

Cluster-Reservierungsmetriken werden als Prozentsatz von CPU, Arbeitsspeicher und GPUs gemessen, die von allen Amazon-ECS-Aufgaben in einem Cluster reserviert sind, im Vergleich zur gesamten CPU- und Speicherauslastung und den GPUs, die für die einzelnen aktiven Container-Instances im Cluster registriert wurden. Nur Container-Instances mit dem Status `ACTIVE` oder `DRAINING` wirken sich auf Metriken zur Cluster-Reservierung aus. Diese Metrik wird nur für Cluster verwendet, deren Aufgaben oder Dienste auf EC2-Instances gehostet werden. Sie wird nicht auf Clustern unterstützt, auf denen Aufgaben gehostet werden. AWS Fargate

```
                              (Total CPU units reserved by tasks in cluster) x 100
Cluster CPU reservation =
 ---------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                 (Total MiB of memory reserved by tasks in cluster x
 100)
Cluster memory reservation =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

```
                                (Total GPUs reserved by tasks in cluster x 100)
Cluster GPU reservation =
 ------------------------------------------------------------------
                              (Total GPUs registered by container instances in cluster)
```

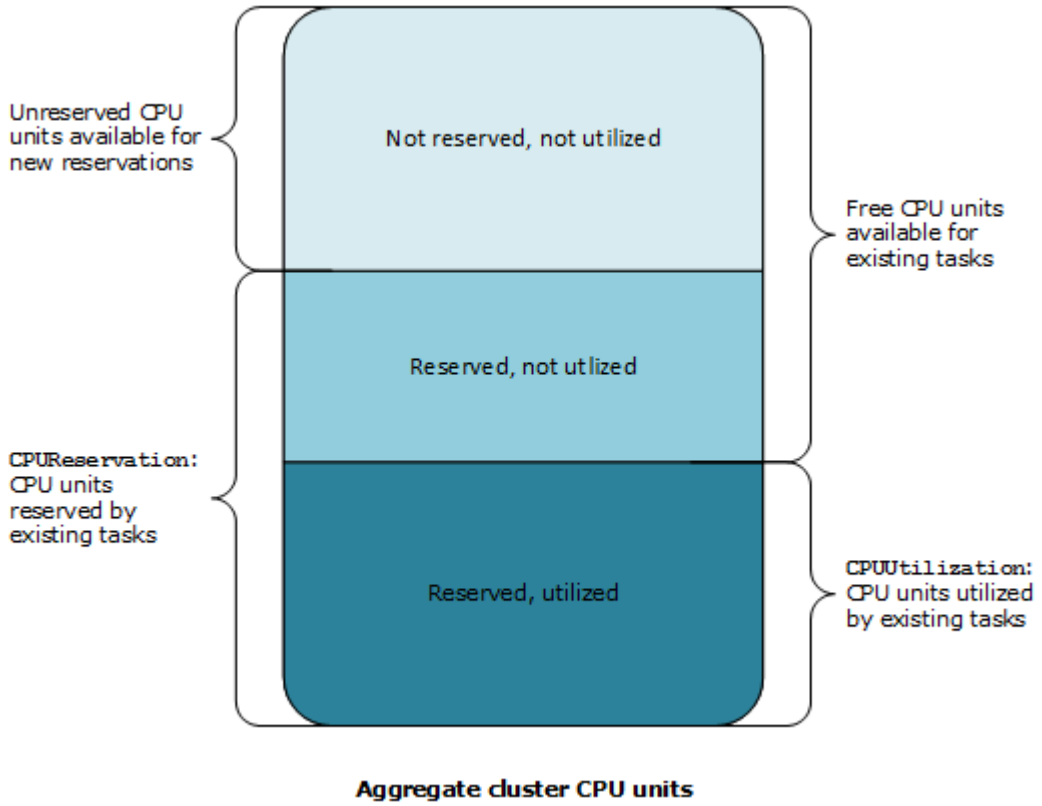
Wenn Sie eine Aufgabe in einem Cluster ausführen, parst Amazon ECS deren Aufgabendefinition und reserviert die aggregierten CPU-Einheiten, MiB Arbeitsspeicher und GPUs, die in ihren Containerdefinitionen angegeben sind. Amazon ECS berechnet jede Minute die Anzahl an CPU-Einheiten, MiB Arbeitsspeicher und GPUs, die aktuell für jede Aufgabe reserviert sind, die im Cluster ausgeführt wird. Die Gesamtmenge an CPU, Arbeitsspeicher und GPUs, die für alle Aufgaben

reserviert sind, die auf dem Cluster ausgeführt werden, wird berechnet, und diese Zahlen werden CloudWatch als Prozentsatz der gesamten registrierten Ressourcen für den Cluster angegeben. Wenn Sie in der Aufgabendefinition eine weiche Grenze (`memoryReservation`) angeben, wird diese verwendet, um die Menge des reservierten Speichers zu berechnen. Andernfalls wird ein hartes Limit (`memory`) verwendet. Der gesamte von Tasks in einem Cluster reservierte MiB-Speicher umfasst auch die Volumengröße des temporären Dateisystems (`tmpfs`) und `sharedMemorySize`, falls in der Taskdefinition definiert. Weitere Informationen über harte und weiche Grenzen, die Größe des gemeinsamen Speichers und die Größe des tmpfs-Volumes finden Sie unter Aufgabendefinitionsparameter.

Für einen Cluster sind beispielsweise zwei aktive Container-Instances registriert: eine `c4.4xlarge`-Instance und eine `c4.large`-Instance. Die `c4.4xlarge`-Instance wird im Cluster mit 16.384 CPU-Einheiten und 30.158 MiB Arbeitsspeicher registriert. Die `c4.large`-Instance wird im Cluster mit 2 048 CPU-Einheiten und 3 768 MiB Arbeitsspeicher registriert. Die aggregierten Ressourcen dieses Clusters sind 18.432 CPU-Einheiten und 33.926 MiB Arbeitsspeicher.

Wenn in einer Aufgabendefinition 1 024 CPU-Einheiten und 2 048 MiB Arbeitsspeicher reserviert sind und zehn Aufgaben mit dieser Aufgabendefinition in diesem Cluster gestartet werden (und aktuell keine anderen Aufgaben ausgeführt werden), werden insgesamt 10 240 CPU-Einheiten und 20 480 MiB Arbeitsspeicher reserviert. Dies entspricht einer CPU-Reservierung von 55% und einer Speicherreservierung von 60% für den Cluster. CloudWatch

Die folgende Abbildung zeigt die gesamten registrierten CPU-Einheiten in einem Cluster und was ihre Reservierung und Nutzung für die bestehenden Aufgaben und die Platzierung neuer Aufgaben bedeutet. Die unteren Blöcke (Reserviert, verwendet) und Mitte (Reserviert, nicht verwendet) stellen die Gesamtzahl der CPU-Einheiten dar, die für die vorhandenen Aufgaben reserviert sind, die auf dem Cluster ausgeführt werden, oder die `CPUReservation` CloudWatch Metrik. Der untere Block steht für die reservierten CPU-Einheiten, die die laufenden Aufgaben tatsächlich auf dem Cluster verwenden, oder für die `CPUUtilization` CloudWatch Metrik. Der obere Block stellt die CPU-Einheiten dar, die nicht für bestehende Aufgaben reserviert sind; diese CPU-Einheiten sind für die Platzierung neuer Aufgaben verfügbar. Bestehende Aufgaben können diese nicht reservierten CPU-Einheiten ebenfalls nutzen, wenn ihr Bedarf an CPU-Ressourcen steigt. Weitere Informationen finden Sie in der `cpu`-Aufgabendefinition der Parameterdokumentation.



Kennzahlen zur Nutzung von Amazon ECS-Clustern

Die Kennzahlen zur Cluster-Auslastung sind für CPU, Arbeitsspeicher und, wenn Ihren Aufgaben ein EBS-Volume zugeordnet ist, für die EBS-Dateisystemauslastung verfügbar. Diese Metriken sind nur für Cluster mit Aufgaben oder Services verfügbar, die auf Amazon EC2 EC2-Instances gehostet werden. Sie werden auf Clustern, auf denen Aufgaben gehostet werden, nicht unterstützt. AWS Fargate

Kennzahlen zur CPU- und Speicherauslastung auf Amazon ECS-Clusterebene

Die CPU- und Speicherauslastung wird als Prozentsatz der CPU und des Speichers gemessen, der von allen Aufgaben in einem Cluster verwendet wird, verglichen mit der Summe von CPU und Arbeitsspeicher, die für jede aktive Amazon EC2 EC2-Instance registriert wurden, die für den Cluster registriert wurde. Nur Amazon EC2 EC2-Instances im `DRAINING` Status `ACTIVE` oder wirken sich auf die Cluster-Nutzungsmetriken aus.

```
                                (Total CPU units used by tasks in cluster) x 100
Cluster CPU utilization =
 ------------------------------------------------------------
```

```
                          (Total CPU units registered by container instances in
 cluster)
```

```
                                    (Total MiB of memory used by tasks in cluster x
 100)
Cluster memory utilization =
 ------------------------------------------------------------------
                             (Total MiB of memory registered by container instances in
 cluster)
```

Jede Minute berechnet der Amazon ECS-Container-Agent auf jeder Amazon EC2 EC2-Instance die Anzahl der CPU-Einheiten und MiB Arbeitsspeicher, die derzeit für jede Aufgabe verwendet werden, die auf dieser Amazon EC2 EC2-Instance ausgeführt wird, und diese Informationen werden an Amazon ECS zurückgemeldet. Die Gesamtmenge an CPU und Arbeitsspeicher, die für alle Aufgaben verwendet wird, die auf dem Cluster ausgeführt werden, wird berechnet, und diese Zahlen werden CloudWatch als Prozentsatz der gesamten registrierten Ressourcen für den Cluster angegeben.

In einem Cluster sind beispielsweise zwei aktive Amazon EC2 EC2-Instances registriert, eine `c4.4xlarge` Instance und eine `c4.large` Instance. Die `c4.4xlarge` Instance registriert sich mit `16,384` CPU-Einheiten und `30,158` MiB Arbeitsspeicher im Cluster. Die `c4.large` Instanz registriert sich bei `2,048` CPU-Einheiten und `3,768` MiB Arbeitsspeicher. Die Gesamtressourcen dieses Clusters sind `18,432` CPU-Einheiten und `33,926` MiB Arbeitsspeicher.

Wenn zehn Aufgaben auf diesem Cluster ausgeführt werden und jede Aufgabe `1,024` CPU-Einheiten und `2,048` MiB Arbeitsspeicher verbraucht, werden insgesamt `10,240` CPU-Einheiten und `20,480` MiB Arbeitsspeicher auf dem Cluster verwendet. Dies entspricht Berichten zufolge CloudWatch einer CPU-Auslastung von 55% und einer Speicherauslastung von 60% für den Cluster.

Nutzung des Amazon EBS-Dateisystems auf Amazon ECS-Clusterebene

Die Metrik zur Nutzung des EBS-Dateisystems auf Clusterebene wird als die Gesamtmenge des verwendeten EBS-Dateisystems durch die auf dem Cluster ausgeführten Aufgaben geteilt durch die Gesamtmenge an EBS-Dateisystemspeicher gemessen, die für alle Aufgaben im Cluster zugewiesen wurde.

```
                                      (Total GB of EBS filesystem used by tasks in
 cluster x 100)
Cluster EBS filesystem utilization =
 ------------------------------------------------------------------
```

```
                                        (Total GB of EBS filesystem allocated to tasks
 in cluster)
```

Kennzahlen zur Nutzung des Amazon ECS-Service

Die Kennzahlen zur Servicenutzung sind für CPU, Arbeitsspeicher und, wenn ein EBS-Volume an Ihre Aufgaben angehängt ist, für die EBS-Dateisystemauslastung verfügbar. Die Service Level-Metriken werden für Services unterstützt, deren Aufgaben sowohl auf Amazon EC2 EC2-Instances als auch auf Fargate gehostet werden.

CPU- und Speicherauslastung auf Service-Ebene

Die CPU- und Speicherauslastung wird als Prozentsatz der CPU und des Speichers gemessen, der von den Amazon ECS-Aufgaben verwendet wird, die zu einem Service auf einem Cluster gehören, verglichen mit der CPU und dem Speicher, die in der Aufgabendefinition des Services angegeben sind.

```
                                      (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ------------------------------------------------------------------------------
                           (Total CPU units specified in task definition) x (number of
 tasks in service)
```

```
                                       (Total MiB of memory used by tasks in service) x
 100
Service memory utilization =
 ------------------------------------------------------------------------------
                              (Total MiB of memory specified in task definition) x
 (number of tasks in service)
```

Jede Minute berechnet der Amazon ECS-Container-Agent die Anzahl der CPU-Einheiten und MiB Arbeitsspeicher, die derzeit für jede Aufgabe verwendet werden, die dem Service gehört, und diese Informationen werden an Amazon ECS zurückgemeldet. Die Gesamtmenge an CPU und Arbeitsspeicher, die für alle Aufgaben verwendet werden, die dem Service gehören und auf dem Cluster ausgeführt werden, wird berechnet. Diese Zahlen werden CloudWatch als Prozentsatz der Gesamtressourcen ausgewiesen, die für den Service in der Aufgabendefinition des Services angegeben sind. Wenn Sie ein Soft-Limit (`memoryReservation`) angeben, dann wird es zur Berechnung der Größe des reservierten Speichers verwendet. Andernfalls wird ein hartes Limit

(`memory`) verwendet. Weitere Informationen zu festen und weichen Grenzwerten finden Sie unterAufgabengröße.

Beispielsweise spezifiziert die Aufgabendefinition für einen Service eine Gesamtmenge von 512 CPU-Einheiten und 1.024 MiB Arbeitsspeicher (mit dem `memory`-Parameter als hartes Limit) für alle ihre Container. Der Service hat eine gewünschte Anzahl von 1 laufenden Aufgabe; er wird auf einem Cluster mit 1 `c4.large`-Container-Instance (mit 2.048 CPU-Einheiten und 3.768 MiB Gesamtarbeitsspeicher) ausgeführt, und es gibt keine anderen Aufgaben, die auf dem Cluster ausgeführt werden. Obwohl die Aufgabe 512 CPU-Einheiten angibt, da sie die einzige laufende Aufgabe auf einer Container-Instance mit 2 048 CPU-Einheiten ist, kann sie bis zum Vierfachen der spezifizierten Menge (2 048/512) verwenden. Jedoch ist der spezifizierte Arbeitsspeicher von 1.024 MiB ein hartes Limit, das nicht überschritten werden kann. Daher kann die Service-Speichernutzung in diesem Fall 100 % nicht überschreiten.

Wenn im vorherigen Beispiel das Soft-Limit `memoryReservation` anstelle des Hard-Limit-Parameters `memory` verwendet worden wäre, könnten die Aufgaben des Services bei Bedarf mehr als die angegebenen 1.024 MiB Speicher verbrauchen. In diesem Fall könnte die Arbeitsspeichernutzung des Service 100 % überschreiten.

Wenn Ihre Anwendung für kurze Zeit einen plötzlichen Anstieg der Speicherauslastung verzeichnet, werden Sie keinen Anstieg der Service-Speicherauslastung feststellen, da Amazon ECS jede Minute mehrere Datenpunkte sammelt und sie dann zu einem Datenpunkt zusammenfasst, an den gesendet wird. CloudWatch

Wenn diese Aufgabe während eines Zeitraums CPU-intensive Arbeiten ausführt und alle 2 048 verfügbaren CPU-Einheiten und 512 MiB Arbeitsspeicher nutzt, meldet der Service 400 % CPU- und 50 % Speichernutzung. Wenn die Aufgabe im Leerlauf ist und 128 CPU-Einheiten und 128 MiB Arbeitsspeicher nutzt, meldet der Service 25 % CPU- und 12,5 % Speichernutzung.

> **Note**
>
> In diesem Beispiel geht die CPU-Auslastung nur über 100 %, wenn die CPU-Einheiten auf Containerebene definiert sind. Wenn Sie CPU-Einheiten auf Task-Ebene definieren, geht die Auslastung nicht über das definierte Limit auf Task-Ebene.

Nutzung des EBS-Dateisystems auf Service-Level

Die Nutzung des EBS-Dateisystems auf Service-Level wird als Gesamtmenge des EBS-Dateisystems gemessen, das von den Aufgaben verwendet wird, die zum Dienst gehören, geteilt durch die Gesamtmenge an EBS-Dateisystemspeicher, die allen Aufgaben zugewiesen ist, die zum Dienst gehören.

```
                                    (Total GB of EBS filesystem used by tasks in the
 service x 100)
Service EBS filesystem utilization =
 ------------------------------------------------------------------
                                    (Total GB of EBS filesystem allocated to tasks
 in the service)
```

Service **RUNNING**-Aufgabenzähler

Sie können CloudWatch Metriken verwenden, um die Anzahl der Aufgaben in Ihren Diensten anzuzeigen, die sich im Status befinden. `RUNNING` Sie können beispielsweise einen CloudWatch Alarm für diese Metrik einrichten, der Sie benachrichtigt, wenn die Anzahl der laufenden Aufgaben in Ihrem Service unter einen bestimmten Wert fällt.

Anzahl der **RUNNING** Serviceaufgaben in Amazon ECS CloudWatch Container Insights

Eine Metrik „Anzahl ausgeführter Aufgaben“ (`RunningTaskCount`) ist pro Cluster und pro Service verfügbar, wenn Sie Amazon ECS CloudWatch Container Insights verwenden. Sie können Container Insights für alle neuen Cluster verwenden, die erstellt wurden, indem Sie sich für die `containerInsights` Kontoeinstellung entscheiden, für einzelne Cluster, indem Sie die Cluster-Einstellungen während der Cluster-Erstellung aktivieren, oder für bestehende Cluster, indem Sie die UpdateClusterSettings API verwenden. Von CloudWatch Container Insights gesammelte Metriken werden als benutzerdefinierte Metriken berechnet. Weitere Informationen zur CloudWatch Preisgestaltung finden Sie unter CloudWatchPreisgestaltung.

Informationen zu dieser Metrik finden Sie unter Amazon ECS Container Insights-Metriken im CloudWatch Amazon-Benutzerhandbuch.

Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge

Mit Amazon EventBridge können Sie Ihre AWS Services automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen reagieren. Ereignisse aus AWS Services werden nahezu EventBridge in Echtzeit übermittelt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen durchgeführt werden sollen, wenn sich für ein Ereignis eine Übereinstimmung mit einer Regel ergibt. Zu den Aktionen, die automatisch konfiguriert werden können, gehören folgende:

- Ereignisse zu Protokollgruppen in CloudWatch Logs hinzufügen
- Eine AWS Lambda Funktion aufrufen
- Aufrufen eines Amazon EC2 Run Command
- Weiterleiten des Ereignisses an Amazon Kinesis Data Streams
- Aktivierung einer AWS Step Functions Zustandsmaschine
- Benachrichtigungen zu einem Amazon SNS-Thema oder einer Amazon Simple Queue Service (Amazon SQS)-Warteschlange

Weitere Informationen finden Sie unter Erste Schritte mit Amazon EventBridge im EventBridge Amazon-Benutzerhandbuch.

Sie können Amazon ECS-Ereignisse verwenden EventBridge , um nahezu in Echtzeit Benachrichtigungen über den aktuellen Status Ihrer Amazon ECS-Cluster zu erhalten. Wenn Ihre Aufgaben den Starttyp EC2 verwenden, können Sie den Status der Container-Instances und den aktuellen Status aller Aufgaben auf diesen Container-Instances sehen. Wenn Ihre Aufgaben den Starttyp Fargate verwenden, können Sie den Status der Container-Instances sehen.

Mithilfe von EventBridge Amazon ECS können Sie benutzerdefinierte Scheduler erstellen, die für die Orchestrierung von Aufgaben zwischen Clustern und die Überwachung des Status von Clustern nahezu in Echtzeit verantwortlich sind. Sie können den Planungs- und Überwachungscode eliminieren, der den Amazon ECS-Service kontinuierlich nach Statusänderungen abfragt, und stattdessen Amazon ECS-Statusänderungen asynchron mit einem beliebigen EventBridge Ziel verarbeiten. Zu den AWS Lambda Zielen könnten Amazon Simple Queue Service, Amazon Simple Notification Service oder Amazon Kinesis Data Streams gehören.

Ein Amazon-ECS-Ereignisstrom stellt sicher, dass jedes Ereignis mindestens einmal übertragen wird. Wenn doppelte Ereignisse gesendet werden, liefert das Ereignis genügend Informationen, um Duplikate zu identifizieren. Weitere Informationen finden Sie unter Umgang mit Amazon ECS-Ereignissen.

Ereignisse sind einigermaßen geordnet, sodass einfach festzustellen ist, wann ein Ereignis in Bezug auf andere Ereignisse aufgetreten ist.

Themen

Amazon-ECS-Events

Amazon ECS verfolgt den Status jeder Ihrer Aufgaben und Services. Wenn sich der Status einer Aufgabe oder eines Services ändert, wird ein Ereignis generiert und an Amazon gesendet EventBridge. Diese Ereignisse werden als Aufgabenstatusänderungsereignisse und Service-Aktionsereignisse klassifiziert. Diese Ereignisse und ihre möglichen Ursachen werden in den folgenden Abschnitten genauer beschrieben.

Amazon ECS hat die folgenden Ereignistypen generiert und an sie gesendet EventBridge: Ereignisse zur Änderung des Container-Instance-Status, Ereignisse zur Änderung des Aufgabenstatus, Serviceaktion und Änderung des Status der Servicebereitstellung.

- Änderung des Status der Container-Instance
- Änderung des Aufgabenstatus
- Bereitstellungs-Zustandsänderung
- Aktion des Dienstes

> **Note**
>
> Amazon ECS fügt in Zukunft möglicherweise andere Ereignistypen, -quellen und -details hinzu. Wenn Sie JSON-Ereignisdaten im Code deserialisieren, stellen Sie sicher, dass Ihre Anwendung darauf vorbereitet ist, unbekannte Eigenschaften zu verarbeiten, um Probleme zu vermeiden, wenn diese zusätzlichen Eigenschaften hinzugefügt werden.

In einigen Fällen werden für dieselbe Aktivität mehrere Ereignisse erstellt. Wenn beispielsweise eine Aufgabe auf einer Container-Instance gestartet wird, wird für die neue Aufgabe ein Ereignis zur Änderung des Taskstatus erstellt. Ein Ereignis zur Änderung des Zustands der Container-Instance wird erstellt, um die Änderung der verfügbaren Ressourcen wie CPU, Speicher und verfügbare Ports in der Container-Instance zu berücksichtigen. Ebenso werden beim Beenden einer Container-Instance Ereignisse für die Container-Instance, den Verbindungsstatus des Container-Agents und jede Aufgabe, die auf der Container-Instance lief, erstellt.

Containerstatus- und Aufgabenstatus-Änderungsereignisse enthalten zwei `version`-Felder: eines im Hauptteil des Ereignisses und eines im `detail`-Objekt des Ereignisses. Im Folgenden werden die Unterschiede zwischen diesen beiden Feldern beschrieben:

- Das Feld `version` im Haupttext des Ereignisses ist für alle Ereignisse auf `0` gesetzt. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.
- Das `version`-Feld im `detail`-Objekt des Ereignisses beschreibt die Version der zugehörigen Ressource. Jedes Mal, wenn sich der Status einer Ressource ändert, erhöht sich diese Versionsnummer. Da Ereignisse mehrfach gesendet werden können, können Sie mit diesem Feld doppelte Ereignisse identifizieren. Doppelte Ereignisse haben im `detail`-Objekt die gleiche Version. Wenn Sie Ihre Amazon ECS-Container-Instance und den Aufgabenstatus mit replizieren EventBridge, können Sie die Version einer Ressource, die von den Amazon ECS-APIs gemeldet wurde, mit der Version vergleichen, die EventBridge für die Ressource (innerhalb des `detail` Objekts) gemeldet wurde, um zu überprüfen, ob die Version in Ihrem Event-Stream aktuell ist.

Service-Aktionsereignisse enthalten nur das Feld `version` im Hauptteil.

Weitere Informationen zur Integration von Amazon ECS und EventBridge finden Sie unter Integration von Amazon EventBridge und Amazon ECS.

Ereignisse zur Änderung des Status der Amazon ECS-Container-Instance

Die folgenden Szenarien führen zu Ereignissen, die den Zustand der Container-Instance verändern:

Sie rufen die API-Operationen `StartTask`, `RunTask` oder `StopTask` entweder direkt oder mit der AWS Management Console oder den SDKs auf.

> Das Platzieren oder Stoppen von Aufgaben in einer Container-Instance ändert die verfügbaren Ressourcen in der Container-Instance (z. B. CPU, Arbeitsspeicher und verfügbare Ports).

Der Amazon-ECS-Service-Scheduler startet oder beendet eine Aufgabe.

Das Platzieren oder Stoppen von Aufgaben in einer Container-Instance ändert die verfügbaren Ressourcen in der Container-Instance (z. B. CPU, Arbeitsspeicher und verfügbare Ports).

Der Amazon-ECS-Containeragent ruft die `SubmitTaskStateChange`-API-Operation mit dem Status `STOPPED` für eine Aufgabe mit dem gewünschten Status `RUNNING` auf.

Der Amazon-ECS-Container-Agent überwacht den Status von Aufgaben auf Ihren Container-Instances und meldet alle Statusänderungen. Wenn eine `RUNNING`-Aufgabe auf `STOPPED` umgestellt wird, gibt der Agent die Ressourcen frei, die der gestoppten Aufgabe zugewiesen wurden (z. B. CPU, Arbeitsspeicher und verfügbare Ports).

Sie melden die Container-Instance mit dem `DeregisterContainerInstance` API-Vorgang entweder direkt oder mit den AWS Management Console oder SDKs ab.

Die Abmeldung einer Container-Instance ändert den Status der Container-Instance und den Verbindungsstatus des Amazon-ECS-Container-Agenten.

Eine Aufgabe wurde gestoppt als eine EC2-Instance gestoppt wurde.

Wenn Sie eine Container-Instance stoppen, wechseln die Aufgaben, die darauf ausgeführt werden, in den Status `STOPPED`.

Der Amazon-ECS-Containeragent registriert eine Container-Instance zum ersten Mal.

Wenn der Amazon-ECS-Container-Agent eine Container-Instance zum ersten Mal registriert (beim Start oder wenn diese erstmals manuell ausgeführt wird), wird ein Statusänderungsereignis für die Instance erstellt.

Der Amazon-ECS-Container-Agent stellt eine Verbindung mit Amazon ECS her oder unterbricht sie.

Wenn der Amazon-ECS-Containeragent eine Verbindung mit dem Amazon-ECS-Backend herstellt oder unterbricht, wechselt der `agentConnected`-Status der Container-Instance.

> **Note**
>
> Der Amazon-ECS-Containeragent trennt und verbindet sich im Rahmen seines normalen Betriebs mehrmals pro Stunde, sodass mit Ereignissen zur Agentenverbindung zu rechnen ist. Diese Ereignisse sind kein Hinweis darauf, dass es ein Problem mit dem Containeragenten oder Ihrer Container-Instance gibt.

Sie aktualisieren den Amazon-ECS-Containeragenten auf einer Instance.

Im Container-Instance-Detail ist ein Objekt für die Container-Agenten-Version enthalten. Wenn Sie den Agent aktualisieren, ändern sich diese Versionsinformationen und ein Ereignis wird erstellt.

Example Änderungsereignis des Container-Instance-Status

Die Statusänderungsereignisse der Container-Instance werden im folgenden Format bereitgestellt. Der folgende `detail` Abschnitt ähnelt dem ContainerInstanceObjekt, das von einem DescribeContainerInstance-API-Vorgang in der Amazon Elastic Container Service API-Referenz zurückgegeben wird. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.


```
{
  "version": "0",
  "id": "8952ba83-7be2-4ab5-9c32-6687532d15a2",
  "detail-type": "ECS Container Instance State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2016-12-06T16:41:06Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315"
  ],
  "detail": {
    "agentConnected": true,
    "attributes": [
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
```

```
      },
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
      {
        "name": "com.amazonaws.ecs.capability.ecr-auth"
      },
      {
        "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": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
      }
    ],
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "containerInstanceArn": "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315",
    "ec2InstanceId": "i-f3a8506b",
    "registeredResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 2048
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 3767
      },
```

```
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "remainingResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 1988
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
```

```
    ],
    "status": "ACTIVE",
    "version": 14801,
    "versionInfo": {
      "agentHash": "aebcbca",
      "agentVersion": "1.13.0",
      "dockerVersion": "DockerVersion: 1.11.2"
    },
    "updatedAt": "2016-12-06T16:41:06.991Z"
  }
}
```

Ereignisse zur Änderung des Amazon ECS-Aufgabenstatus

Die folgenden Szenarien verursachen Ereignisse zur Änderung des Aufgabenzustands:

Sie rufen die API-Operationen `StartTask`, `RunTask` oder `StopTask` entweder direkt oder mit der AWS Management Console, AWS CLI oder den SDKs auf.

Das Starten oder Stoppen von Aufgaben erstellt neue Aufgabenressourcen oder ändert den Status bestehender Aufgabenressourcen.

Der Amazon-ECS-Service-Scheduler startet oder beendet eine Aufgabe.

Das Starten oder Stoppen von Aufgaben erstellt neue Aufgabenressourcen oder ändert den Status bestehender Aufgabenressourcen.

Der Amazon-ECS-Containeragent ruft die `SubmitTaskStateChange`-API-Operation auf.

Für den Amazon EC2 EC2-Starttyp überwacht der Amazon ECS-Container-Agent den Status Ihrer Aufgaben auf Ihren Container-Instances. Der Amazon ECS-Container-Agent meldet alle Statusänderungen. Statusänderungen können Änderungen von `PENDING` auf `RUNNING` oder von `RUNNING` auf `STOPPED` beinhalten.

Sie erzwingen die Abmeldung der zugrunde liegenden Container-Instance mit dem `DeregisterContainerInstance` API-Vorgang und dem `force` Flag, entweder direkt oder mit den AWS Management Console oder SDKs.

Die Abmeldung einer Container-Instance ändert den Status der Container-Instance und den Verbindungsstatus des Amazon-ECS-Container-Agenten. Wenn Aufgaben auf der Container-Instance ausgeführt werden, muss das `force`-Flag so festgelegt sein, dass eine Abmeldung zulässig ist. Damit werden alle Aufgaben auf der Instance gestoppt.

Die zugrundeliegende Container-Instance wird gestoppt oder beendet.

Wenn Sie eine Container-Instance stoppen oder beenden, wechseln die Aufgaben, die darauf ausgeführt werden, in den Status `STOPPED`.

Ein Container in der Aufgabe ändert den Status.

Der Amazon-ECS-Container-Agent überwacht den Status der Container innerhalb von Aufgaben. Wenn zum Beispiel ein Container, der innerhalb einer Aufgabe läuft, anhält, wird durch diese Änderung des Containerzustands ein Ereignis erstellt.

Eine Aufgabe, die den Fargate Spot-Kapazitätsanbieter nutzt, erhält eine Beendigungsankündigung.

Wenn eine Aufgabe den `FARGATE_SPOT`-Kapazitätsanbieter nutzt und aufgrund einer Spot-Unterbrechung gestoppt wird, wird ein Ereignis zur Änderung des Aufgabenstatus erstellt.

Example Änderungsereignis des Aufgabenstatus

Aufgabenstatusänderungsereignisse werden im folgenden Format bereitgestellt. Der folgende `detail` Abschnitt ähnelt dem Task-Objekt, das von einem DescribeTasksAPI-Vorgang in der Amazon Elastic Container Service API-Referenz zurückgegeben wird. Wenn Ihre Container ein mit Amazon ECR gehostetes Image verwenden, wird das Feld `imageDigest` zurückgegeben.

> **Note**
>
> Die Werte für die Felder `createdAt`, `connectivityAt`, `pullStartedAt`, `startedAt`, `pullStoppedAt` und `updatedAt` sind in der Antwort einer `DescribeTasks`-Aktion UNIX-Zeitstempel. Im Aufgabenstatus-Änderungsereignis sind sie jedoch ISO-Zeichenfolgenzeitstempel.

Weitere Informationen zu CloudWatch Veranstaltungsparametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.

Informationen zur Konfiguration einer EventBridge Amazon-Ereignisregel, die nur Aufgabenereignisse erfasst, bei denen die Ausführung der Aufgabe beendet wurde, weil einer ihrer wichtigsten Container beendet wurde, finden Sie unter Senden von Amazon Simple Notification Service-Benachrichtigungen für Ereignisse, bei denen Amazon ECS-Aufgaben gestoppt wurden

```
{
    "version": "0",
```

```
    "id": "3317b2af-7005-947d-b652-f55e762e571a",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-01-23T17:57:58Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad"
    ],
    "detail": {
        "attachments": [
            {
                "id": "1789bcae-ddfb-4d10-8ebe-8ac87ddba5b8",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-abcd1234"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-abcd1234"
                    },
                    {
                        "name": "macAddress",
                        "value": "0a:98:eb:a7:29:ba"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.0.139"
                    }
                ]
            }
        ],
        "availabilityZone": "us-west-2c",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/FargateCluster",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-west-2:111122223333:container/
cf159fd6-3e3f-4a9e-84f9-66cbe726af01",
                "lastStatus": "RUNNING",
                "name": "FargateApp",
```

```
                "image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/hello-
repository:latest",
                "imageDigest":
 "sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6",
                "runtimeId":
 "ad64cbc71c7fb31c55507ec24c9f77947132b03d48d9961115cf24f3b7307e1e",
                "taskArn": "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad",
                "networkInterfaces": [
                    {
                        "attachmentId": "1789bcae-ddfb-4d10-8ebe-8ac87ddba5b8",
                        "privateIpv4Address": "10.0.0.139"
                    }
                ],
                "cpu": "0"
            }
        ],
        "createdAt": "2020-01-23T17:57:34.402Z",
        "launchType": "FARGATE",
        "cpu": "256",
        "memory": "512",
        "desiredStatus": "RUNNING",
        "group": "family:sample-fargate",
        "lastStatus": "RUNNING",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "FargateApp"
                }
            ]
        },
        "connectivity": "CONNECTED",
        "connectivityAt": "2020-01-23T17:57:38.453Z",
        "pullStartedAt": "2020-01-23T17:57:52.103Z",
        "startedAt": "2020-01-23T17:57:58.103Z",
        "pullStoppedAt": "2020-01-23T17:57:55.103Z",
        "updatedAt": "2020-01-23T17:57:58.103Z",
        "taskArn": "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad",
        "taskDefinitionArn": "arn:aws:ecs:us-west-2:111122223333:task-definition/
sample-fargate:1",
        "version": 4,
        "platformVersion": "1.3.0"
    }
```

```
}
```

Aktionsereignisse des Amazon ECS-Service

Amazon ECS sendet Service-Aktionsereignisse mit dem Detailtyp ECS Service Action (ECS-Service-Aktion). Im Gegensatz zu den Container-Instance- und Aufgabenstatus-Änderungsereignissen enthalten die Ereignisse der Service-Aktion keine Versionsnummer im `details`-Antwortfeld. Das Folgende ist ein Ereignismuster, das verwendet wird, um eine EventBridge Regel für Amazon ECS-Serviceaktionsereignisse zu erstellen. Weitere Informationen finden Sie unter EventBridgeRegel erstellen im EventBridge Amazon-Benutzerhandbuch.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Service Action"
    ]
}
```

Amazon ECS sendet Ereignisse mit den Ereignistypen `INFO`, `WARN` und `ERROR`. Im Folgenden finden Sie die Ereignisse der Service-Aktion.

Service-Aktionsereignisse mit **`INFO`**-Ereignistyp

`SERVICE_STEADY_STATE`

Der Service befindet sich in einem korrekten Status und umfasst die gewünschte Anzahl von Aufgaben. Somit erreicht er einen stabilen Status. Der Service-Scheduler meldet den Status regelmäßig, sodass Sie diese Nachricht möglicherweise mehrmals erhalten.

`TASKSET_STEADY_STATE`

Der Aufgabensatz ist gesund und in der gewünschten Anzahl von Aufgaben und erreicht so einen Steady-Status.

`CAPACITY_PROVIDER_STEADY_STATE`

Ein Kapazitätsanbieter, der einem Service zugeordnet ist, erreicht einen Steady-Status.

`SERVICE_DESIRED_COUNT_UPDATED`

Wenn der Service-Scheduler die berechnete gewünschte Anzahl für einen Service oder einen Aufgabensatz aktualisiert. Dieses Ereignis wird nicht gesendet, wenn die gewünschte Anzahl manuell von einem Benutzer aktualisiert wird.

Service-Aktionsereignisse mit **`WARN`**-Ereignistyp

`SERVICE_TASK_START_IMPAIRED`

Der Service kann Aufgaben nicht konsistent und erfolgreich starten.

`SERVICE_DISCOVERY_INSTANCE_UNHEALTHY`

Ein Service, der die Serviceerkennung verwendet, enthält eine fehlerhafte Aufgabe. Der Service-Scheduler erkennt, dass eine Aufgabe innerhalb einer Service-Registrierung ungesund ist.

Service-Aktionsereignisse mit **`ERROR`**-Ereignistyp

`SERVICE_DAEMON_PLACEMENT_CONSTRAINT_VIOLATED`

Eine Aufgabe in einem Service, der die `DAEMON`-Service-Scheduler-Strategie verwendet, entspricht nicht mehr der Platzierungsbeschränkungsstrategie für den Service.

`ECS_OPERATION_THROTTLED`

Der Service-Scheduler wurde aufgrund der Amazon-ECS-API-Drosselgrenzen gedrosselt.

`SERVICE_DISCOVERY_OPERATION_THROTTLED`

Der Service Scheduler wurde aufgrund der AWS Cloud Map API-Drosselungsgrenzen gedrosselt. Dies kann bei Services auftreten, die für die Verwendung der Serviceerkennung konfiguriert sind.

`SERVICE_TASK_PLACEMENT_FAILURE`

Der Service-Scheduler kann keinen Task platzieren. Die Ursache wird im Feld `reason` beschrieben.

Eine typische Ursache für die Erstellung dieses Service-Ereignisses ist ein Mangel an Ressourcen in dem Cluster, in dem die Aufgabe platziert werden soll. Beispiele sind unzureichende CPU- oder Speicherkapazität in den verfügbaren Container-Instances oder die Tatsache, dass keine Container-Instances verfügbar sind. Eine weitere typische Ursache besteht darin, dass der

Amazon-ECS-Container-Agent in der Container-Instance getrennt wird, sodass der Scheduler die Aufgabe nicht platzieren kann.

`SERVICE_TASK_CONFIGURATION_FAILURE`

Der Service-Scheduler kann aufgrund eines Konfigurationsfehlers keine Aufgabe platzieren. Die Ursache wird im Feld `reason` beschrieben.

Eine häufige Ursache für die Erstellung dieses Service-Ereignisses ist, dass Tags auf den Service angewendet wurden, der Benutzer oder die Rolle aber nicht für das neue Amazon Ressourcenname (ARN)-Format in der Region optiert hatte. Weitere Informationen finden Sie unter Amazon Ressourcennamen (ARNs) und IDs. Eine weitere typische Ursache besteht darin, dass Amazon ECS die bereitgestellte IAM-Aufgabenrolle nicht annehmen konnte.

Example Service-Steady-Statusereignis

Service-Steady-Statusereignisse werden im folgenden Format bereitgestellt. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.

```
{
    "version": "0",
    "id": "af3c496d-f4a8-65d1-70f4-a69d52e9b584",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:27:22Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:27:22.695Z"
    }
}
```

Example Kapazitätsanbieter-Steady-Statusereignis

Die Steady-Statusereignisse des Kapazitätsanbieters werden im folgenden Format bereitgestellt.

```
{
    "version": "0",
    "id": "b9baa007-2f33-0eb1-5760-0d02a572d81f",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:37:00Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "CAPACITY_PROVIDER_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/ASG-tutorial-
capacity-provider"
        ],
        "createdAt": "2019-11-19T19:37:00.807Z"
    }
}
```

Example Service-Aufgabenstart-Gefährdet-Ereignis

Service-Aufgabenstart-Gefährdet-Ereignisse werden im folgenden Format bereitgestellt.

```
{
    "version": "0",
    "id": "57c9506e-9d21-294c-d2fe-e8738da7e67d",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "WARN",
        "eventName": "SERVICE_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:55:38.725Z"
```

```
    }
}
```

Example Service-Aufgabenplazierung-Fehlerereignis

Fehlerereignisse bei der Platzierung von Service-Aufgaben werden im folgenden Format bereitgestellt. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.

Im folgenden Beispiel versuchte die Aufgabe, den `FARGATE_SPOT`-Kapazitätsanbieter zu verwenden, aber der Service-Scheduler konnte keine Fargate Spot-Kapazität akquirieren.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/FARGATE_SPOT"
        ],
        "reason": "RESOURCE:FARGATE",
        "createdAt": "2019-11-06T19:09:33.087Z"
    }
}
```

Im folgenden Beispiel für den EC2-Launchtyp wurde versucht, die Aufgabe auf der Container-Instance `2dd1b186f39845a584488d2ef155c131` zu launchen, aber der Service-Scheduler konnte die Aufgabe wegen unzureichender CPU nicht platzieren.

```
{
  "version": "0",
```

```
  "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
  "detail-type": "ECS Service Action",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2019-11-19T19:55:38Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
  ],
  "detail": {
    "eventType": "ERROR",
    "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
    "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
    "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
    ],
    "reason": "RESOURCE:CPU",
    "createdAt": "2019-11-06T19:09:33.087Z"
  }
}
```

Ereignisse zur Änderung des Bereitstellungsstatus des Amazon ECS-Service

Amazon ECS sendet Änderungsereignisse für die Servicebereitstellung mit dem Detailtyp Statusänderungen des ECS-Bereitstellungszustands. Das Folgende ist ein Ereignismuster, das verwendet wird, um eine EventBridge Regel für Ereignisse zur Änderung des Bereitstellungsstatus von Amazon ECS-Services zu erstellen. Weitere Informationen finden Sie unter EventBridgeRegel erstellen im EventBridge Amazon-Benutzerhandbuch.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Deployment State Change"
    ]
}
```

Amazon ECS sendet Ereignisse mit den Ereignistypen `INFO` und `ERROR`. Folgende sind die Änderungsereignisse für den Bereitstellungsstatus von Services:

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Die Dienstbereitstellung ist in Bearbeitung. Dieses Ereignis wird sowohl für Erstbereitstellungen als auch für Rollbackbereitstellungen gesendet.

`SERVICE_DEPLOYMENT_COMPLETED`

Die Dienstbereitstellung ist abgeschlossen. Dieses Ereignis wird gesendet, sobald ein Service einen Steady-Status nach einer Bereitstellung erreicht.

`SERVICE_DEPLOYMENT_FAILED`

Die Dienstbereitstellung ist fehlgeschlagen. Dieses Ereignis wird für Dienste gesendet, bei denen die Logik des Bereitstellungsschutzschalters eingeschaltet ist.

Example Ereignis „Dienstbereitstellung in Bearbeitung“

Ereignisse zur Servicebereitstellung werden bereitgestellt, wenn sowohl eine anfängliche als auch eine Rollbackbereitstellung gestartet wird. Der Unterschied zwischen den beiden liegt im Feld `reason`. Weitere Informationen zu EventBridge Parametern finden Sie unter Ereignisse und Ereignismuster im EventBridge Amazon-Benutzerhandbuch.

Im Folgenden sehen Sie eine Beispielausgabe für eine erste Bereitstellung, die gestartet wird.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3a6EXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment deploymentId in progress."
   }
```

```
}
```

Im Folgenden sehen Sie eine Beispielausgabe für eine Rollbackbereitstellung, die gestartet wird. Das Feld `reason` enthält die ID der Bereitstellung, auf die der Service zurückgesetzt wird.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment circuit breaker: rolling back to
 deploymentId deploymentID."
   }
}
```

Example Ereignis „Dienstbereitstellung abgeschlossen“

„Dienstbereitstellung abgeschlossen“-Statusereignisse werden im folgenden Format bereitgestellt. Weitere Informationen finden Sie unter Stellen Sie Amazon ECS-Services bereit, indem Sie Aufgaben ersetzen.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
```

```
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_COMPLETED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment deploymentID completed."
    }
}
```

Example Ereignis „Dienstbereitstellung ist fehlgeschlagen“

„Dienstbereitstellung fehlgeschlagen“-Statusereignisse werden im folgenden Format bereitgestellt. Ein Dienstbereitstellungsstatusereignis wird nur für Dienste gesendet, für die die Logik des Bereitstellungsschutzschalters eingeschaltet ist. Weitere Informationen finden Sie unter Stellen Sie Amazon ECS-Services bereit, indem Sie Aufgaben ersetzen.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "ERROR",
         "eventName": "SERVICE_DEPLOYMENT_FAILED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment circuit breaker: task failed to start."
    }
}
```

Umgang mit Amazon ECS-Ereignissen

Amazon ECS sendet Ereignisse mindestens einmal. Das bedeutet, dass Sie möglicherweise mehrere Kopien eines bestimmten Ereignisses erhalten. Außerdem werden Ereignisse eventuell nicht in der Reihenfolge, in der sie stattgefunden haben, an Ihre Ereignis-Listener übermittelt.

Damit Ereignisse richtig geordnet werden können, enthält der `detail`-Abschnitt der einzelnen Ereignisse eine Eigenschaft`version`. Jedes Mal, wenn sich der Status einer Ressource ändert, erhöht sich diese `version`. Doppelte Ereignisse haben im `detail`-Objekt die gleiche `version`. Wenn Sie Ihre Amazon ECS-Container-Instance und den Aufgabenstatus mit replizieren EventBridge, können Sie die von den Amazon ECS-APIs gemeldete Version einer Ressource mit der EventBridge für die Ressource `version` gemeldeten Version vergleichen, um zu überprüfen, ob die Version in Ihrem Event-Stream aktuell ist. Ereignisse mit einer höheren Versionseigenschaftsnummer sind später einzuordnen, als Ereignisse mit niedrigeren Versionsnummern.

Beispiel: Behandlung von Ereignissen in einer AWS Lambda -Funktion

Das folgende Beispiel zeigt eine in Python 3.9 geschriebene Lambda-Funktion, die sowohl Statusänderungsereignisse von Aufgaben als auch von Container-Instance-Zustand erfasst und in einer von zwei Amazon-DynamoDB-Tabellen speichert:

- CtrInstanceECS-Status — Speichert den neuesten Status einer Container-Instance. Die Tabellen-ID ist der `containerInstanceArn`-Wert der Container-Instance.
- ECS TaskState — Speichert den neuesten Status einer Aufgabe. Die Tabellen-ID ist der `taskArn`-Wert der Aufgabe.

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
```

```
        event_id = event["detail"]["taskArn"]
    elif event["detail-type"] == "ECS Container Instance State Change":
        table_name = "ECSCtrInstanceState"
        id_name =  "containerInstanceArn"
        event_id = event["detail"]["containerInstanceArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))
```

```
        table.put_item(
            Item=new_record
        )
```

Das folgende Fargate-Beispiel zeigt eine in Python 3.9 geschriebene Lambda-Funktion, die Ereignisse zur Änderung des Aufgabenstatus erfasst und in der folgenden Amazon DynamoDB-Tabelle speichert:

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
 without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")
```

```
    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

Überwachen Sie Amazon ECS-Container mit Container Insights

CloudWatch Container Insights sammelt, aggregiert und fasst Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zusammen.

Container Insights erkennt alle laufenden Container in einem Cluster und sammelt Leistungsdaten auf jeder Ebene des Performance-Stacks. Betriebliche Daten werden als Performance-Protokollereignisse gesammelt. Dabei handelt es sich um Einträge mit einem strukturierten JSON-Schema für die Aufnahme und Speicherung von Daten hoher Kardinalität in in großem Umfang. Aus diesen Daten werden übergeordnete aggregierte Metriken auf Cluster-, Service- und Aufgabenebene als Metriken CloudWatch erstellt. CloudWatch Die Metriken umfassen die Auslastung für Ressourcen

wie z B. CPU, Arbeitsspeicher, Datenträger und Netzwerk. Die Metriken sind in CloudWatch automatischen Dashboards verfügbar. Informationen zu den verfügbaren Metriken finden Sie unter Amazon ECS Container Insights-Metriken im CloudWatch Amazon-Benutzerhandbuch.

> **Important**
>
> Von CloudWatch Container Insights gesammelte Metriken werden als benutzerdefinierte Metriken berechnet. Weitere Informationen zur CloudWatch Preisgestaltung finden Sie unter CloudWatchPreisgestaltung. Amazon ECS stellt außerdem Überwachungsmetriken bereit, die ohne zusätzliche Gebühren bereitgestellt werden. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS mit CloudWatch .

Überlegungen

Folgendes sollte bei der Verwendung von CloudWatch Container Insights beachtet werden.

- CloudWatch Die Container Insights-Metriken spiegeln nur die Ressourcen wider, für die innerhalb des angegebenen Zeitraums Aufgaben ausgeführt wurden. Wenn Sie beispielsweise einen Cluster mit einem Dienst haben, dieser Dienst jedoch keine Aufgaben in einem bestimmten `RUNNING` Status hat, werden keine Metriken an diese gesendet CloudWatch. Wenn Sie über zwei Services verfügen, und einer von diesen hat laufende Aufgaben und der andere nicht, werden nur die Metriken für den Service mit laufenden Aufgaben gesendet.
- Netzwerkmetriken sind für alle Aufgaben verfügbar, die auf Fargate ausgeführt werden und für Aufgaben, die auf Amazon-EC2-Instances ausgeführt werden, die entweder den `bridge`- oder den `awsvpc`-Netzwerkmodus verwenden.

Sie können Amazon ECS-Aufgaben- und Service-Lifecycle-Ereignisse in der CloudWatch Container Insights-Konsole anzeigen. Auf diese Weise können Sie Ihre Container-Metriken, Protokolle und Ereignisse in einer einzigen Ansicht korrelieren und Sie erhalten einen umfassenderen Einblick in die Funktionalität.

Die Ereignisse, die Sie anzeigen können, sind diejenigen, die Amazon ECS an Amazon sendet EventBridge. Weitere Informationen finden Sie unter Amazon-ECS-Ereignisse.

Sie können Leistungsmetriken für Cluster, Aufgaben oder Services konfigurieren. Abhängig von der ausgewählten Ressource werden die folgenden Ereignisse gemeldet:

- Änderungsereignisse des Container-Instance-Status
- Service-Aktionsereignisse
- Änderungsereignisse des Aufgabenstatus

Konfiguration von CloudWatch Container Insights für Amazon ECS

Sie können Container Insights mithilfe der Amazon ECS-Konsole AWS CLI, der API und der SDKs konfigurieren.

Verwenden Sie die folgende Tabelle, um die Aktion zu bestimmen, die für das Hinzufügen von Container Insights ergriffen werden muss.

Markierungsunterstützung für Amazon-ECS-Ressourcen

Aufgabe	Konsole	AWS CLI	API-Aktion
Ändern Sie die Standardeinstellung für alle Benutzer	Amazon ECS-Konto einstellungen ändern	put-account-setting-default	PutAccountSettingDefault
Ändern Sie die Standardeinstellung für einen bestimmten Benutzer	Amazon ECS-Konto einstellungen ändern	Put-Account-Einstellung	PutAccountEinstellung
Container Insights für einen bestimmten Cluster konfigurieren	Erstellen eines Amazon ECS-Clusters für den Starttyp Fargate Erstellen eines Amazon ECS-Clusters für den Amazon EC2 EC2-Starttyp Aktualisierung eines Amazon ECS-Clusters	create-cluster UpdateCluster	CreateCluster UpdateCluster

> **Important**
>
> Für Cluster mit Aufgaben oder Services, die den EC2-Starttyp verwenden, müssen Ihre Container-Instances Version 1.29.0 oder höher des Amazon-ECS-Agenten ausführen. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.

Erforderliche Berechtigungen für CloudWatch Container Insights zum Anzeigen von Amazon ECS-Lebenszyklusereignissen

Sie müssen die richtigen Berechtigungen konfigurieren und dann können Sie die Ereignisse in der CloudWatch Container Insights-Konsole konfigurieren und anzeigen. Weitere Informationen finden Sie unter Amazon ECS-Lebenszyklusereignisse in Container Insights im CloudWatch Amazon-Benutzerhandbuch. Weitere Informationen zu IAM-Richtlinien für finden Sie CloudWatch unter AWS Identity and Access Management für CloudWatch.

Erforderliche Berechtigungen zum Konfigurieren von Container Insights zur Anzeige von Lebenszyklusereignissen von Amazon ECS

Die folgenden Berechtigungen sind für die Aufgabenrolle erforderlich, um die Lebenszyklusereignisse zu konfigurieren:

- Ereignisse: PutRule
- Ereignisse: PutTargets
- Protokolle: CreateLog Gruppe

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
```

```
      "Resource": "*"
    }
  ]
}
```

Erforderliche Berechtigungen zum Anzeigen von Lebenszyklusereignissen von Amazon ECS in Container Insights

Die folgenden Berechtigungen sind erforderlich, um die Lebenszyklusereignisse anzuzeigen. Fügen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur Aufgabenausführungsrolle hinzu. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- Ereignisse: DescribeRule
- Ereignisse: ListTargets ByRule
- Protokolle: DescribeLog Gruppen

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Ermitteln Sie den Zustand von Amazon ECS-Aufgaben mithilfe von Container-Zustandsprüfungen

Wenn Sie eine Aufgabendefinition erstellen, können Sie eine Integritätsprüfung für Ihre Container konfigurieren. Integritätsprüfungen sind Befehle, die lokal in einem Container ausgeführt werden und den Zustand und die Verfügbarkeit von Anwendungen überprüfen.

Der Amazon-ECS-Container-Agent überwacht die in der Aufgabendefinition angegebenen Zustandsprüfungen und erstellt Berichte darüber. Amazon ECS überwacht keine Docker-Zustandsprüfungen, die in ein Container-Image eingebettet, aber nicht in der Container-Definition angegeben sind. Parameter für die Zustandsprüfung, die in einer Container-Definition angegeben sind, überschreiben Docker-Zustandsprüfungen, die im Container-Image vorhanden sind.

Wenn in einer Aufgabendefinition eine Integritätsprüfung definiert ist, führt der Container die Integritätsprüfung innerhalb des Containers aus und wertet dann den Exit-Code aus, um den Zustand der Anwendung zu ermitteln.

Die Integritätsprüfung besteht aus den folgenden Parametern:

- Befehl — Der Befehl, den der Container ausführt, um festzustellen, ob er fehlerfrei ist. Das Zeichenfolge-Array kann mit `CMD` beginnen, um die Befehlsargumente direkt auszuführen, oder mit `CMD-SHELL`, um den Befehl mit der Standard-Shell des Containers auszuführen.
- Intervall — Der Zeitraum (in Sekunden) zwischen den einzelnen Zustandsprüfungen.
- Timeout — Der Zeitraum (in Sekunden), in dem auf eine erfolgreiche Zustandsprüfung gewartet werden soll, bevor sie als Fehlschlag gewertet wird.
- Wiederholungen — Gibt an, wie oft eine fehlgeschlagene Zustandsprüfung wiederholt werden muss, bevor der Container als fehlerhaft eingestuft wird.
- Startzeitraum — Die optionale Übergangszeit, um Containern Zeit zum Bootstrapping zu geben, bevor fehlgeschlagene Integritätsprüfungen auf die maximale Anzahl von Wiederholungen angerechnet werden.

Informationen zum Angeben einer Integritätsprüfung in einer Aufgabendefinition finden Sie unter. Zustandsprüfung

Im Folgenden werden die möglichen Integritätsstatuswerte für einen Container beschrieben:

- `HEALTHY` – Die Container-Zustandsprüfung wurde erfolgreich bestanden.
- `UNHEALTHY` – Die Container-Zustandsprüfung ist fehlgeschlagen.
- `UNKNOWN` – Die Container-Zustandsprüfung wird ausgewertet, es ist keine Container-Zustandsprüfung definiert oder Amazon ECS verfügt nicht über den Zustandstatus des Containers.

Die Befehle zur Integritätsprüfung werden auf dem Container ausgeführt. Daher müssen Sie die Befehle in das Container-Image aufnehmen.

Die Integritätsprüfung stellt über die Loopback-Schnittstelle des Containers unter `localhost` oder eine Verbindung zur Anwendung her. `127.0.0.1` Ein Exit-Code von `0` bedeutet Erfolg, und ein Exit-Code ungleich Null zeigt einen Fehler an.

Beachten Sie bei der Verwendung von Container-Integritätsprüfungen Folgendes:

- Container Zustandsprüfungen erfordern Version 1.17.0 oder höher des Amazon-ECS-Container-Agenten.
- Container-Zustandsprüfungen werden für Fargate-Aufgaben unterstützt, wenn Sie eine Linux-Plattformversion `1.1.0` oder höher oder eine Windows-Plattformversion `1.1.0` oder höher verwenden

Wie Amazon ECS den Zustand von Aufgaben bestimmt

Container, die unverzichtbar sind und in der Aufgabendefinition den Befehl Health Check enthalten, sind die einzigen, die bei der Bestimmung des Aufgabenzustands berücksichtigt werden.

Die folgenden Regeln werden der Reihe nach ausgewertet:

1. Wenn der Status eines wichtigen Containers lautet`UNHEALTHY`, dann ist der Aufgabenstatus`UNHEALTHY`.
2. Wenn der Status eines wichtigen Containers ist`UNKNOWN`, dann ist der Aufgabenstatus`UNKNOWN`.
3. Wenn der Status aller wichtigen Container lautet`HEALTHY`, dann ist der Aufgabenstatus`HEALTHY`.

Betrachten Sie das folgende Beispiel für den Zustand einer Aufgabe mit zwei wichtigen Containern.

Zustand von Container 1	Behälter 2 Gesundheit	Gesundheit der Aufgabe
UNHEALTHY	UNKNOWN	UNHEALTHY
UNHEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY

Betrachten Sie das folgende Beispiel für den Zustand einer Aufgabe mit 3 Containern.

Zustand von Container 1	Behälter 2 Gesundheit	Behälter 3 Gesundheit	Gesundheit der Aufgabe
UNHEALTHY	UNKNOWN	UNKNOWN	UNHEALTHY
UNHEALTHY	UNKNOWN	HEALTHY	UNHEALTHY
UNHEALTHY	HEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	HEALTHY	UNKNOWN
HEALTHY	UNKNOWN	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY	HEALTHY

Wie wirken sich Verbindungsabbrüche auf Integritätsprüfungen von Agenten aus

Wenn der Amazon ECS-Container-Agent vom Amazon ECS-Service getrennt wird, führt dies nicht dazu, dass ein Container in einen UNHEALTHY Status wechselt. Dies ist beabsichtigt, um sicherzustellen, dass Container auch bei Neustarts der Agenten oder bei vorübergehender Nichtverfügbarkeit weiterlaufen. Der Status der Zustandsprüfung ist die Antwort „Zuletzt gehört von“ des Amazon ECS-Agenten. Wenn der Container also HEALTHY vor dem Trennen berücksichtigt wurde, bleibt dieser Status bestehen, bis der Agent wieder eine Verbindung herstellt und eine weitere Zustandsprüfung erfolgt. Es werden keine Annahmen über den Status der Zustandsprüfungen der Container getroffen.

Zustand des Amazon ECS-Containers anzeigen

Sie können den Zustand des Containers in der Konsole und mithilfe der API in der `DescribeTasks` Antwort einsehen. Weitere Informationen finden Sie DescribeTasksin der Amazon Elastic Container Service API-Referenz.

Wenn Sie die Protokollierung für Ihren Container verwenden, zum Beispiel Amazon CloudWatch Logs, können Sie den Health Check-Befehl so konfigurieren, dass die Ausgabe des Container-Zustands an Ihre Logs weitergeleitet wird. Stellen Sie sicher, dass `2&1` Sie sowohl catch als auch die `stdout` `stderr` Informationen verwenden.

```
"command": [
     "CMD-SHELL",
     "curl -f http://localhost/ >> /proc/1/fd/1 2>&1  || exit 1"
   ],
```

Überwachen Sie den Zustand der Amazon ECS-Container-Instance

Amazon ECS bietet die Zustandsüberwachung von Container-Instances an. Sie können schnell ermitteln, ob Amazon ECS Probleme erkannt hat, die für Ihre Container-Instances möglicherweise das Ausführen von Containern verhindern. Amazon ECS führt automatische Prüfungen an jeder laufenden Container-Instance mit Agent-Version `1.57.0` oder höher durch, um Probleme zu identifizieren. Weitere Informationen zum Überprüfen der Agent-Version einer Container-Instance finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Sie müssen AWS CLI Version `1.22.3` oder höher oder AWS CLI Version `2.3.6` oder höher verwenden. Informationen zur Aktualisierung von finden Sie unter Installation oder Aktualisierung der neuesten Version von AWS CLI im AWS Command Line Interface Benutzerhandbuch Version 2. AWS CLI

Statusprüfungen werden etwa zweimal pro Minute durchgeführt und geben als Status „Bestanden“ oder „Fehler“ zurück. Wenn alle Überprüfungen bestanden wurden, lautet der Gesamtstatus der Instance `OK`. Falls mindestens eine Überprüfung nicht bestanden wird, lautet der Gesamtstatus `IMPAIRED`. Statusprüfungen sind in den Amazon-ECS-Container-Agent integriert und können daher nicht deaktiviert oder gelöscht werden. Sie können die Ergebnisse dieser Statusprüfungen anzeigen, um bestimmte bzw. erkennbare Probleme zu ermitteln. Weitere Informationen finden Sie unter the section called “Zustandsprüfung”.

Führen Sie die `DescribeContainerInstances` API mit der `CONTAINER_INSTANCE_HEALTH` Option aus, den Zustand der Container-Instance abzurufen.

```
aws ecs describe-container-instances \
     --cluster cluster_name \
     --container-instances 47279cd2cadb41cbaef2dcEXAMPLE \
     --include CONTAINER_INSTANCE_HEALTH
```

Im Folgenden finden Sie ein Beispiel für das Zustandsstatus-Objekt in der Ausgabe.

```
"healthStatus": {
```

```
 "overallStatus": "OK",
 "details": [{
  "type": "CONTAINER_RUNTIME",
  "status": "OK",
  "lastUpdated": "2021-11-10T03:30:26+00:00",
  "lastStatusChange": "2021-11-10T03:26:41+00:00"
 }]
}
```

Verwandte Themen

- Überwachen Sie Amazon ECS mit CloudWatch

Identifizieren Sie Amazon ECS-Optimierungsmöglichkeiten mithilfe von Anwendungs-Trace-Daten

Amazon ECS ist in AWS Distro integriert OpenTelemetry , um Trace-Daten aus Ihrer Anwendung zu sammeln. Amazon ECS verwendet einen Container „ AWS Distro for OpenTelemetry Sidecar“, um Trace-Daten zu sammeln und an diese weiterzuleiten. AWS X-Ray Weitere Informationen finden Sie unter AWS Distro for OpenTelemetry Collector in Amazon ECS einrichten. Anschließend können Sie Fehler und Ausnahmen identifizieren, Leistungsengpässe und Reaktionszeiten analysieren. AWS X-Ray

Damit die AWS Distribution for OpenTelemetry Collector Trace-Daten senden kann, muss Ihre Anwendung so konfiguriert sein AWS X-Ray, dass sie die Trace-Daten erstellt. Weitere Informationen finden Sie unter Instrumentieren Ihrer Anwendung für AWS X-Ray im Entwicklerhandbuch zu AWS X-Ray .

Erforderliche IAM-Berechtigungen für AWS Distro für die Integration mit OpenTelemetry AWS X-Ray

Die Amazon ECS-Integration mit AWS Distro for OpenTelemetry erfordert, dass Sie eine Aufgabenrolle erstellen und die Rolle in Ihrer Aufgabendefinition angeben. Wir empfehlen Ihnen, die AWS Distro for OpenTelemetry Sidecar so zu konfigurieren, dass Container-Logs an Logs weitergeleitet werden. CloudWatch

> **Important**
>
> Wenn Sie auch Anwendungsmetriken mithilfe der AWS Distro für die OpenTelemetry Integration erfassen, stellen Sie sicher, dass Ihre Task-IAM-Rolle auch die für diese Integration erforderlichen Berechtigungen enthält. Weitere Informationen finden Sie unter Korrelieren Sie die Amazon ECS-Anwendungsleistung mithilfe von Anwendungsmetriken.

Erstellen Sie die folgende Richtlinie und fügen Sie sie dann der Aufgabenausführungsrolle hinzu.

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy",
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries",
                "ssm:GetParameters"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Geben Sie die AWS Distribution für OpenTelemetry Sidecar zur AWS X-Ray Integration in Ihre Aufgabendefinition an

Die Amazon ECS-Konsole vereinfacht die Erstellung des Containers AWS Distro for OpenTelemetry Sidecar mithilfe der Option Trace-Erfassung verwenden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Wenn Sie die Amazon ECS-Konsole nicht verwenden, können Sie den Container AWS Distro for OpenTelemetry Sidecar zu Ihrer Aufgabendefinition hinzufügen. Der folgende Ausschnitt aus der Aufgabendefinition zeigt die Container-Definition für das Hinzufügen des AWS Distro for Sidecar zur Integration. OpenTelemetry AWS X-Ray

```
{
 "family": "otel-using-xray",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryXrayRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
```

```
    "image": "application-image",
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "true",
      "awslogs-group": "/ecs/aws-otel-emitter",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
     }
    },
    "dependsOn": [{
     "containerName": "aws-otel-collector",
     "condition": "START"
    }]
   },
   {
    "name": "aws-otel-collector",
    "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
    "essential": true,
    "command": [
     "--config=/etc/ecs/otel-instance-metrics-config.yaml"
    ],
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "True",
      "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
     }
    }
   }
  ],
  "networkMode": "awsvpc",
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Korrelieren Sie die Amazon ECS-Anwendungsleistung mithilfe von Anwendungsmetriken

Amazon ECS on Fargate unterstützt das Sammeln von Metriken aus Ihren auf Fargate ausgeführten Anwendungen und deren Export nach Amazon CloudWatch oder Amazon Managed Service for Prometheus.

Sie können die gesammelten Metadaten verwenden, um Leistungsdaten von Anwendungen mit den zugrunde liegenden Infrastrukturdaten zu korrelieren und so die durchschnittliche Zeit bis zur Lösung des Problems zu reduzieren.

Amazon ECS verwendet einen Container „ AWS Distro for OpenTelemetry Sidecar“, um Ihre Anwendungsmetriken zu sammeln und an das Ziel weiterzuleiten. Die Amazon ECS-Konsole vereinfacht das Hinzufügen dieser Integration bei der Erstellung Ihrer Aufgabendefinitionen.

Themen

Anwendungsmetriken nach Amazon exportieren CloudWatch

Amazon ECS on Fargate unterstützt den Export Ihrer benutzerdefinierten Anwendungsmetriken CloudWatch als benutzerdefinierte Metriken nach Amazon. Dazu fügen Sie Ihrer Aufgabendefinition den Container AWS Distro for OpenTelemetry Sidecar hinzu. Die Amazon ECS-Konsole vereinfacht diesen Vorgang, indem sie beim Erstellen einer neuen Aufgabendefinition die Option Metrikerfassung verwenden hinzufügt. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Die Anwendungsmetriken werden in CloudWatch Logs mit dem Namen der Protokollgruppe exportiert, `/aws/ecs/application/metrics` und die Metriken können im `ECS/AWSOTel/Application` Namespace eingesehen werden. Ihre Anwendung muss mit dem SDK instrumentiert sein. OpenTelemetry Weitere Informationen finden Sie unter Einführung in AWS Distro OpenTelemetry in der Dokumentation zu AWS Distro. OpenTelemetry

Überlegungen

Folgendes sollte beachtet werden, wenn Sie die Amazon ECS on Fargate-Integration mit AWS Distro verwenden, OpenTelemetry um Anwendungsmetriken an Amazon zu senden. CloudWatch

- Diese Integration sendet nur Ihre benutzerdefinierten Anwendungsmetriken an. CloudWatch Wenn Sie Metriken auf Aufgabenebene wünschen, können Sie Container Insights in der Amazon-ECS-Cluster-Konfiguration aktivieren. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.
- Die AWS Distribution for OpenTelemetry Integration wird für Amazon ECS-Workloads, die auf Fargate gehostet werden, und Amazon ECS-Workloads, die auf Amazon EC2 EC2-Instances gehostet werden, unterstützt. Externe Instances werden derzeit nicht unterstützt.
- CloudWatch unterstützt maximal 30 Dimensionen pro Metrik. Standardmäßig enthält Amazon ECS die Dimensionen `TaskARN`, `ClusterARN`, `LaunchType`, `TaskDefinitionFamily` und `TaskDefinitionRevision` in den Metriken. Die restlichen 25 Dimensionen können durch Ihre Anwendung definiert werden. Wenn mehr als 30 Dimensionen konfiguriert sind, CloudWatch können sie nicht angezeigt werden. In diesem Fall werden die Anwendungsmetriken im `ECS/AWSOTel/Application` CloudWatch Metrik-Namespace angezeigt, jedoch ohne Dimensionen. Sie können Ihre Anwendung so einstellen, dass zusätzliche Dimensionen hinzugefügt werden. Weitere Informationen finden Sie unter Using CloudWatch metrics with AWS Distro for OpenTelemetry in der Dokumentation zu AWS Distro. OpenTelemetry

Erforderliche IAM-Berechtigungen für AWS Distro für die OpenTelemetry Integration mit Amazon CloudWatch

Die Amazon ECS-Integration mit AWS Distro for OpenTelemetry erfordert, dass Sie eine Task-IAM-Rolle erstellen und die Rolle in Ihrer Aufgabendefinition angeben. Wir empfehlen, dass AWS Distro for OpenTelemetry Sidecar auch so konfiguriert wird, dass Container-Protokolle an Logs weitergeleitet werden. Dazu muss auch eine IAM-Rolle für die Aufgabenausführung erstellt und in Ihrer Aufgabendefinition angegeben werden. CloudWatch Die Amazon ECS-Konsole kümmert sich in Ihrem Namen um die IAM-Rolle für die Aufgabenausführung, aber die Aufgaben-IAM-Rolle muss manuell erstellt und zu Ihrer Aufgabendefinition hinzugefügt werden. Weitere Informationen zur IAM-Aufgabenausführungsrolle finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

> **Important**
>
> Wenn Sie auch Anwendungs-Trace-Daten mithilfe der AWS Distro für die OpenTelemetry Integration sammeln, stellen Sie sicher, dass Ihre Task-IAM-Rolle auch die für diese Integration erforderlichen Berechtigungen enthält. Weitere Informationen finden Sie unter Identifizieren Sie Amazon ECS-Optimierungsmöglichkeiten mithilfe von Anwendungs-Trace-Daten.

Wenn Ihre Anwendung zusätzliche Berechtigungen benötigt, sollten Sie diese dieser Richtlinie hinzufügen. Jede Aufgabendefinition darf nur eine Aufgaben-IAM-Rolle angeben. Wenn Sie beispielsweise eine in Systems Manager gespeicherte benutzerdefinierte Konfigurationsdatei verwenden, sollten Sie dieser IAM-Richtlinie die `ssm:GetParameters`-Berechtigung hinzufügen.

So erstellen Sie die Servicerolle für Elastic Container Service (IAM-Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
4. Wählen Sie für Service oder Anwendungsfall Elastic Container Service und dann den Anwendungsfall Elastic Container Service Task aus.
5. Wählen Sie Weiter aus.
6. Suchen Sie im Bereich Berechtigungen hinzufügen nach der Richtlinie AWSDistroOpenTelemetryPolicyForXrayund wählen Sie sie dann aus.
7. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature, das für Servicerollen verfügbar ist, aber nicht für servicegebundene Rollen.

   a. Öffnen Sie den Abschnitt Berechtigungsgrenze festlegen und wählen Sie dann Eine Berechtigungsgrenze verwenden aus, um die maximalen Rollenberechtigungen zu steuern.

      IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto.

   b. Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.
8. Wählen Sie Weiter aus.
9. Geben Sie einen Rollennamen oder ein Rollennamensuffix ein, damit Sie den Zweck der Rolle leichter erkennen können.

   Important

   Beachten Sie beim Benennen einer Rolle Folgendes:

- Rollennamen müssen innerhalb Ihres AWS-Konto Unternehmens eindeutig sein und können nicht von Fall zu Fall eindeutig sein.

  Erstellen Sie beispielsweise keine Rollen, die **`PRODROLE`** sowohl als auch benannt sind**`prodrole`**. Wenn ein Rollenname in einer Richtlinie oder als Teil eines ARN verwendet wird, unterscheidet der Rollenname zwischen Groß- und Kleinschreibung. Wenn Kunden jedoch ein Rollenname in der Konsole angezeigt wird, z. B. während des Anmeldevorgangs, wird die Groß- und Kleinschreibung nicht berücksichtigt.
- Sie können den Namen der Rolle nicht bearbeiten, nachdem er erstellt wurde, da andere Entitäten möglicherweise auf die Rolle verweisen.

10. (Optional) Geben Sie unter Beschreibung eine Beschreibung für die Rolle ein.
11. (Optional) Um die Anwendungsfälle und Berechtigungen für die Rolle zu bearbeiten, wählen Sie in den Abschnitten Schritt 1: Vertrauenswürdige Entitäten auswählen oder Schritt 2: Berechtigungen hinzufügen die Option Bearbeiten aus.
12. (Optional) Um die Rolle leichter zu identifizieren, zu organisieren oder nach ihr zu suchen, fügen Sie Tags als Schlüssel-Wert-Paare hinzu. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.
13. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Geben Sie in Ihrer AWS Aufgabendefinition die Distribution für OpenTelemetry Sidecar an

Die Amazon ECS-Konsole vereinfacht die Erstellung des Containers AWS Distro for OpenTelemetry Sidecar mithilfe der Option Metrikerfassung verwenden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Wenn Sie die Amazon ECS-Konsole nicht verwenden, können Sie den Container AWS Distro for OpenTelemetry Sidecar manuell zu Ihrer Aufgabendefinition hinzufügen. Das folgende Beispiel für eine Aufgabendefinition zeigt die Container-Definition für das Hinzufügen der AWS Integration Distro for OpenTelemetry Sidecar for Amazon. CloudWatch

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [
```

```
    {
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-cloudwatch.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Exportieren von Anwendungsmetriken an Amazon Managed Service for Prometheus

Amazon ECS unterstützt den Export Ihrer CPU-, Speicher-, Netzwerk- und Speichermetriken auf Aufgabenebene sowie Ihrer benutzerdefinierten Anwendungsmetriken an Amazon Managed Service for Prometheus. Dazu fügen Sie Ihrer Aufgabendefinition den Container AWS Distro for OpenTelemetry Sidecar hinzu. Die Amazon ECS-Konsole vereinfacht diesen Vorgang, indem sie beim Erstellen einer neuen Aufgabendefinition die Option Metrikerfassung verwenden hinzufügt. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Die Metriken werden an Amazon Managed Service for Prometheus exportiert und können über das Dashboard von Amazon Managed Grafana eingesehen werden. Ihre Anwendung muss entweder mit Prometheus-Bibliotheken oder mit dem SDK instrumentiert sein. OpenTelemetry Weitere Informationen zur Instrumentierung Ihrer Anwendung mit dem OpenTelemetry SDK finden Sie unter Introduction to AWS Distro for OpenTelemetry in der Dokumentation von Distro. AWS OpenTelemetry

Bei Verwendung der Prometheus-Bibliotheken muss Ihre Anwendung einen `/metrics`-Endpunkt bereitstellen, der zum Scraping der Metrikdaten verwendet wird. Weitere Informationen zum Instrumentieren Ihrer Anwendung mit Prometheus-Bibliotheken finden Sie unter Prometheus-Clientbibliotheken in der Prometheus-Dokumentation.

Überlegungen

Folgendes sollte beachtet werden, wenn Sie die Amazon ECS on Fargate-Integration mit AWS Distro verwenden, OpenTelemetry um Anwendungsmetriken an Amazon Managed Service for Prometheus zu senden.

- Die AWS Distribution for OpenTelemetry Integration wird für Amazon ECS-Workloads, die auf Fargate gehostet werden, und Amazon ECS-Workloads, die auf Amazon EC2 EC2-Instances gehostet werden, unterstützt. Externe Instances werden derzeit nicht unterstützt.
- Standardmäßig OpenTelemetry beinhaltet AWS Distro for alle verfügbaren Dimensionen auf Aufgabenebene für Ihre Anwendungsmetriken beim Export nach Amazon Managed Service for Prometheus. Sie können Ihre Anwendung auch so instrumentieren, dass zusätzliche Dimensionen hinzugefügt werden. Weitere Informationen finden Sie in der Dokumentation unter Erste Schritte mit Prometheus Remote Write Exporter for Amazon Managed Service for Prometheus in der Distribution. AWS OpenTelemetry

Erforderliche IAM-Berechtigungen für AWS Distro für die OpenTelemetry Integration mit Amazon Managed Service for Prometheus

Die Amazon ECS-Integration mit Amazon Managed Service for Prometheus unter Verwendung von AWS Distro for OpenTelemetry Sidecar erfordert, dass Sie eine Task-IAM-Rolle erstellen und die Rolle in Ihrer Aufgabendefinition angeben. Diese Aufgaben-IAM-Rolle muss manuell mit den folgenden Schritten erstellt werden, bevor Sie Ihre Aufgabendefinition anmelden.

Wir empfehlen, dass AWS Distro for OpenTelemetry Sidecar auch so konfiguriert wird, dass Container-Protokolle an Logs weitergeleitet werden. Dazu muss auch eine IAM-Rolle für die Aufgabenausführung erstellt und in Ihrer Aufgabendefinition angegeben werden. CloudWatch Die Amazon ECS-Konsole kümmert sich in Ihrem Namen um die IAM-Rolle für die Aufgabenausführung, die Aufgaben-IAM-Rolle muss jedoch manuell erstellt werden. Weitere Informationen zum Erstellen der IAM-Aufgabenausführungsrolle finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

> **Important**
>
> Wenn Sie auch Anwendungs-Trace-Daten mithilfe der AWS Distro für die OpenTelemetry Integration sammeln, stellen Sie sicher, dass Ihre Task-IAM-Rolle auch die für diese Integration erforderlichen Berechtigungen enthält. Weitere Informationen finden Sie unter Identifizieren Sie Amazon ECS-Optimierungsmöglichkeiten mithilfe von Anwendungs-Trace-Daten.

So erstellen Sie die Servicerolle für Elastic Container Service (IAM-Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
4. Wählen Sie für Service oder Anwendungsfall Elastic Container Service und dann den Anwendungsfall Elastic Container Service Task aus.
5. Wählen Sie Weiter aus.
6. Suchen Sie im Bereich Berechtigungen hinzufügen nach AmazonPrometheusRemoteWriteAccess und wählen Sie dann die Richtlinie aus.

7. (Optional) Legen Sie eine Berechtigungsgrenze fest. Dies ist ein erweitertes Feature, das für Servicerollen verfügbar ist, aber nicht für servicegebundene Rollen.

   a. Öffnen Sie den Abschnitt Berechtigungsgrenze festlegen und wählen Sie dann Eine Berechtigungsgrenze verwenden aus, um die maximalen Rollenberechtigungen zu steuern.

      IAM enthält eine Liste der AWS verwalteten und kundenverwalteten Richtlinien in Ihrem Konto.

   b. Wählen Sie die Richtlinie aus, die für eine Berechtigungsgrenze verwendet werden soll.

8. Wählen Sie Weiter aus.

9. Geben Sie einen Rollennamen oder ein Rollennamensuffix ein, damit Sie den Zweck der Rolle leichter erkennen können.

   > **Important**
   >
   > Beachten Sie beim Benennen einer Rolle Folgendes:
   >
   > - Rollennamen müssen innerhalb Ihres AWS-Konto Unternehmens eindeutig sein und können nicht von Fall zu Fall eindeutig sein.
   >
   >   Erstellen Sie beispielsweise keine Rollen, die **`PRODROLE`** sowohl als auch benannt sind**`prodrole`**. Wenn ein Rollenname in einer Richtlinie oder als Teil eines ARN verwendet wird, unterscheidet der Rollenname zwischen Groß- und Kleinschreibung. Wenn Kunden jedoch ein Rollenname in der Konsole angezeigt wird, z. B. während des Anmeldevorgangs, wird die Groß- und Kleinschreibung nicht berücksichtigt.
   > - Sie können den Namen der Rolle nicht bearbeiten, nachdem er erstellt wurde, da andere Entitäten möglicherweise auf die Rolle verweisen.

10. (Optional) Geben Sie unter Beschreibung eine Beschreibung für die Rolle ein.

11. (Optional) Um die Anwendungsfälle und Berechtigungen für die Rolle zu bearbeiten, wählen Sie in den Abschnitten Schritt 1: Vertrauenswürdige Entitäten auswählen oder Schritt 2: Berechtigungen hinzufügen die Option Bearbeiten aus.

12. (Optional) Um die Rolle leichter zu identifizieren, zu organisieren oder nach ihr zu suchen, fügen Sie Tags als Schlüssel-Wert-Paare hinzu. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Markieren von IAM-Ressourcen im IAM-Benutzerhandbuch.

13. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

Geben Sie in Ihrer AWS Aufgabendefinition die Distribution für OpenTelemetry Sidecar an

Die Amazon ECS-Konsole vereinfacht die Erstellung des Containers AWS Distro for OpenTelemetry Sidecar mithilfe der Option Metrikerfassung verwenden. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.

Wenn Sie die Amazon ECS-Konsole nicht verwenden, können Sie den Container AWS Distro for OpenTelemetry Sidecar manuell zu Ihrer Aufgabendefinition hinzufügen. Das folgende Beispiel für eine Aufgabendefinition zeigt die Container-Definition für das Hinzufügen der Integration AWS Distro for OpenTelemetry Sidecar für Amazon Managed Service for Prometheus.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-amp.yaml"
   ],
   "environment": [{
    "name": "AWS_PROMETHEUS_ENDPOINT",
```

```
    "value": "https://aps-workspaces.aws-region.amazonaws.com/workspaces/
ws-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/api/v1/remote_write"
   }],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Amazon ECS-API-Aufrufe protokollieren mit AWS CloudTrail

Amazon ECS ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon ECS ausgeführt wurden. CloudTrail erfasst alle API-Aufrufe für Amazon ECS als Ereignisse, einschließlich Aufrufe von der Amazon ECS-Konsole und von Codeaufrufen an die Amazon ECS-API-Operationen. Um Ihre VPC zu schützen, werden Anfragen, die durch eine VPC-Endpunktrichtlinie abgelehnt wurden, aber andernfalls zugelassen worden wären, nicht aufgezeichnet. CloudTrail

Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für Amazon ECS. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon ECS gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.

Amazon ECS-Informationen in CloudTrail

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn eine Aktivität in Amazon ECS auftritt, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto ansehen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung von Ereignissen in Ihrem AWS Konto, einschließlich Ereignissen für Amazon ECS, erstellen Sie einen Trail, der zur Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket CloudTrail verwendet wird. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails
- CloudTrail Unterstützte Dienste und Integrationen
- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

Alle Amazon ECS-Aktionen werden von der Amazon Elastic Container Service API-Referenz protokolliert CloudTrail und sind dort dokumentiert. Beispielsweise generieren Aufrufe der `DeleteCluster` AbschnitteCreateService, `RunTask` und Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Anmeldeinformationen des Root-Benutzers oder des Benutzers gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter dem CloudTrail UserIdentity-Element.

Erläuterungen der Amazon-ECS-Protokolldateieinträge

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

> **Note**
>
> Diese Beispiele wurden für eine bessere Lesbarkeit formatiert. In einer CloudTrail Protokolldatei sind alle Einträge und Ereignisse in einer einzigen Zeile zusammengefasst. Außerdem ist dieses Beispiel auf einen einzigen Amazon-ECS-Eintrag beschränkt. In einer echten CloudTrail Protokolldatei sehen Sie Einträge und Ereignisse von mehreren Diensten. AWS

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `CreateCluster` Aktion demonstriert:

```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
        "arn": "arn:aws:sts::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-20T18:32:25Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Mary_Major"
```

```
            }
        }
    },
    "eventTime": "2018-06-20T19:04:36Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clusterName": "default"
    },
    "responseElements": {
        "cluster": {
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "pendingTasksCount": 0,
            "registeredContainerInstancesCount": 0,
            "status": "ACTIVE",
            "runningTasksCount": 0,
            "statistics": [],
            "clusterName": "default",
            "activeServicesCount": 0
        }
    },
    "requestID": "cb8c167e-EXAMPLE",
    "eventID": "e3c6f4ce-EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Überwachen Sie Workloads mithilfe von Amazon ECS-Metadaten

Sie können die Aufgaben- und Container-Metadaten verwenden, um Fehler bei Ihren Workloads zu beheben und Konfigurationsänderungen auf der Grundlage der Laufzeitumgebung vorzunehmen.

Metadaten umfassen die folgenden Kategorien:

- Attribute auf Taskebene, die Informationen darüber liefern, wo die Aufgabe ausgeführt wird.
- Attribute auf Containerebene, die die Docker-ID, den Namen und die Bilddetails bereitstellen.

  Dies bietet Einblick in den Container.
- Netzwerkeinstellungen wie IP-Adressen, Subnetze und Netzwerkmodus.

Dies hilft bei der Netzwerkkonfiguration und Fehlerbehebung.

- Status und Zustand der Aufgabe

  So wissen Sie, ob die Aufgaben ausgeführt werden.

Sie können Metadaten mit einer der folgenden Methoden anzeigen:

- Container-Metadatendatei

  Ab Version 1.15.0 des Amazon-ECS-Container-Agenten stehen verschiedene Container-Metadaten in den Containern oder der Host-Container-Instance zur Verfügung. Wenn Sie dieses Feature aktivieren, können Sie die Informationen zu einer Aufgabe, einem Container und einer Container-Instance aus dem Container oder der Host-Container-Instance abrufen. Die Metadaten-Datei wird auf der Host-Instance erstellt und im Container als Docker-Volume bereitgestellt. Diese Datei ist daher nicht verfügbar, wenn eine Aufgabe in AWS Fargate gehostet wird.
- Endpunkt für Aufgabenmetadaten

  Der Amazon-ECS-Container-Agent injiziert in jeden Container eine Umgebungsvariable, die als Endpunkt für Task-Metadaten benannt wird, die dem Container verschiedene Task-Metadaten und Docker-Statistiken bereitstellt.
- Container-Introspektion

  Der Amazon-ECS-Container-Agent bietet eine API-Operation für das Erfassen von Details über die Container-Instance, auf der der Agent ausgeführt wird, sowie über die zugewiesenen Aufgaben, die auf dieser Instance ausgeführt werden.

Amazon ECS Container-Metadatendatei

Ab Version 1.15.0 des Amazon-ECS-Container-Agenten stehen verschiedene Container-Metadaten in den Containern oder der Host-Container-Instance zur Verfügung. Wenn Sie dieses Feature aktivieren, können Sie die Informationen zu einer Aufgabe, einem Container und einer Container-Instance aus dem Container oder der Host-Container-Instance abrufen. Die Metadatendatei wird auf der Host-Instance erstellt und im Container als Docker-Volume bereitgestellt. Sie ist daher nicht verfügbar, wenn eine Aufgabe auf AWS Fargate gehostet wird.

Die Container-Metadatendatei wird auf der Host-Instance bereinigt, wenn der Container bereinigt wird. Mit der Container-Agent-Variablen `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` können

Sie anpassen, wann dies geschieht. Weitere Informationen finden Sie unter Automatische Amazon ECS-Aufgabe und -Image-Bereinigung.

Themen

Speicherorte der Container-Metadatendatei

Standardmäßig wird die Container-Metadatendatei unter den folgenden Host- und Container-Pfaden abgelegt.

- Für Linux-Instances:
  - Host-Pfad: `/var/lib/ecs/data/metadata/`*`cluster_name`*`/`*`task_id`*`/`*`container_name`*`/ecs-container-metadata.json`

    > **Note**
    >
    > Der Linux-Host-Pfad geht davon aus, dass der Standardinstallationspfad des Datenverzeichnisses verwendet wird (`/var/lib/ecs/data`), wenn der Agent gestartet wird. Wenn Sie nicht das Amazon-ECS-optimierte AMI verwenden (oder das `ecs-init`-Paket, um den Container-Agenten zu starten und zu verwalten), stellen Sie sicher, dass die Konfigurationsvariable `ECS_HOST_DATA_DIR` des Agenten auf den Host-Pfad gesetzt ist, unter dem sich die Statusdatei des Container-Agenten befindet. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.
  - Container-Pfad: `/opt/ecs/metadata/`*`random_ID`*`/ecs-container-metadata.json`
- Für Windows-Instances:
  - Host-Pfad: `C:\ProgramData\Amazon\ECS\data\metadata\`*`task_id`*`\`*`container_name`*`\ecs-container-metadata.json`
  - Container-Pfad: `C:\ProgramData\Amazon\ECS\metadata\`*`random_ID`*`\ecs-container-metadata.json`

Der Speicherort der Container-Metadatendatei wird jedoch auf die Umgebungsvariable `ECS_CONTAINER_METADATA_FILE` im Container gesetzt, um einen einfachen Zugriff zu gestatten. Sie können den Dateihalt mit dem folgenden Befehl aus dem Container heraus lesen:

- Für Linux-Instances:

```
cat $ECS_CONTAINER_METADATA_FILE
```

- Für Windows-Instanzen ()PowerShell:

```
Get-Content -path $env:ECS_CONTAINER_METADATA_FILE
```

Amazon ECS-Container-Metadaten aktivieren

Sie können Container-Metadaten auf Container-Instance-Ebene aktivieren, indem Sie die `ECS_ENABLE_CONTAINER_METADATA`-Variable des Container-Agenten auf `true` festlegen. Sie können diese Variable in der `/etc/ecs/ecs.config` Konfigurationsdatei festlegen und den Agenten neu starten. Sie können sie auch zur Laufzeit beim Starten des Agentencontainers als Docker-Umgebungsvariable festlegen. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Wenn für `ECS_ENABLE_CONTAINER_METADATA` beim Start des Agenten `true` eingestellt ist, werden Metadatendateien für alle Container erstellt, die ab diesem Zeitpunkt erstellt werden. Der Amazon-ECS-Container-Agent kann keine Metadaten-Dateien für Container erstellen, die erstellt wurden, bevor die Container-Agent-Variable `ECS_ENABLE_CONTAINER_METADATA` auf `true` gesetzt wurde. Um sicherzustellen, dass alle Container Metadatendateien empfangen, sollten Sie diese Agentenvariable beim Start der Container-Instance setzen. Folgendes ist ein Beispiel für ein Benutzerdatenskript, das diese Variable festlegt und die Container-Instance beim Cluster registriert.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_ENABLE_CONTAINER_METADATA=true
EOF
```

Dateiformat für Amazon ECS-Container-Metadaten

Die folgenden Informationen werden in der JSON-Datei mit den Container-Metadaten abgelegt.

`Cluster`

Der Name des Clusters, in dem die Aufgabe des Containers ausgeführt wird

`ContainerInstanceARN`

Der vollständige Amazon Resource Name (ARN) der Host-Container-Instance.

`TaskARN`

Der vollständige Amazon Resource Name (ARN) der Aufgabe, zu der der Container gehört.

`TaskDefinitionFamily`

Der Name der Aufgabendefinitionsfamilie, die der Container verwendet.

`TaskDefinitionRevision`

Die Aufgabendefinitionsrevision, die der Container verwendet.

`ContainerID`

Die Docker-Container-ID (und nicht die Amazon ECS Container-ID) für den Container.

`ContainerName`

Der Container-Name aus der Amazon-ECS-Aufgabendefinition für den Container.

`DockerContainerName`

Der Container-Name, den der Docker-Daemon für den Container verwendet (z. B. der Name, der in der Ausgabe des Befehls docker ps angezeigt wird)

`ImageID`

Der SHA-Digest für das Docker-Image zum Starten des Containers.

`ImageName`

Der Image-Name und das Tag für das Docker-Image zum Starten des Containers.

`PortMappings`

Alle Port-Zuordnungen für den Container.

`ContainerPort`

Der Port, auf dem der Container offengelegt ist.

`HostPort`

Der Port, auf der Host-Container-Instance, der offengelegt ist.

`BindIp`

Die Binde-IP-Adresse, die dem Container von Docker zugewiesen wurde. Diese IP-Adresse wird nur mit dem Netzwerkmodus `bridge` angewendet und steht nur über die Container-Instance zur Verfügung.

`Protocol`

Das für die Port-Zuweisung verwendete Netzwerkprotokoll.

`Networks`

Der Netzwerk-Modus und die IP-Adresse für den Container.

`NetworkMode`

Der Netzwerk-Modus für die Aufgabe, zu der der Container gehört.

`IPv4Addresses`

Die IP-Adressen, die mit dem Container verknüpft sind.

> **Important**
>
> Wenn Ihre Aufgabe den Netzwerkmodus `awsvpc` verwendet, wird die IP-Adresse des Containers nicht zurückgegeben. In diesem Fall können Sie die IP-Adresse abrufen, indem Sie die Datei /etc/hosts mit dem folgenden Befehl auslesen:
>
> ```
> tail -1 /etc/hosts | awk '{print $1}'
> ```

`MetadataFileStatus`

Der Status der Metadatendatei. Wenn der Status `READY` ist, ist die Metadatendatei aktuell und vollständig. Wenn die Datei noch nicht bereit ist (z. B. in dem Moment, in dem die Aufgabe gestartet wird), steht eine gekürzte Version des Dateiformats zur Verfügung. Um eine Wettlaufsituation zu vermeiden, wenn der Container gestartet wurde, die Metadaten aber noch nicht geschrieben wurden, können Sie die Metadatendatei analysieren und warten, bis dieser Parameter auf `READY` gesetzt wurde, bevor Sie die Metadaten verwenden. Diese stehen in der Regel in weniger als 1 Sekunde zur Verfügung, nachdem der Container gestartet wurde.

`AvailabilityZone`

Die Availability Zone, in der sich die Host-Container-Instance befindet.

`HostPrivateIPv4Address`

Die private IP-Adresse der Aufgabe, zu der der Container gehört.

`HostPublicIPv4Address`

Die öffentliche IP-Adresse der Aufgabe, zu der der Container gehört.

Example Amazon ECS Container-Metadatendatei (**`READY`**)

Das folgende Beispiel zeigt eine Container-Metadatendatei im Status `READY`.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/2b88376d-
aba3-4950-9ddf-bcb0f388a40c",
    "TaskDefinitionFamily": "console-sample-app-static",
    "TaskDefinitionRevision": "1",
    "ContainerID": "aec2557997f4eed9b280c2efd7afccdcedfda4ac399f7480cae870cfc7e163fd",
    "ContainerName": "simple-app",
    "CreatedAt": "2023-10-08T20:09:11.44527186Z",
    "StartedAt": "2023-10-08T20:09:11.44527186Z",
    "DockerContainerName": "/ecs-console-sample-app-static-1-simple-app-
e4e8e495e8baa5de1a00",
    "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de",
    "ImageName": "httpd:2.4",
    "PortMappings": [
        {
            "ContainerPort": 80,
            "HostPort": 80,
            "BindIp": "0.0.0.0",
            "Protocol": "tcp"
        }
    ],
    "Networks": [
        {
            "NetworkMode": "bridge",
```

```
            "IPv4Addresses": ["192.0.2.0"]
        }
    ],
    "MetadataFileStatus": "READY",
    "AvailabilityZone": "us-east-1b",
    "HostPrivateIPv4Address": "192.0.2.0",
    "HostPublicIPv4Address": "203.0.113.0"
}
```

Example Unvollständige Amazon ECS Container-Metadatendatei (noch nicht **`READY`**)

Das folgende Beispiel zeigt eine Container-Metadatendatei, die den Status `READY` noch nicht erreicht hat. Die Informationen in der Datei sind auf ein paar wenige Parameter beschränkt, die aus der Aufgabendefinition bekannt sind. Die Container-Metadatendatei sollte innerhalb von 1 Sekunde nach dem Starten des Containers zur Verfügung stehen.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/
d90675f8-1a98-444b-805b-3d9cabb6fcd4",
    "ContainerName": "metadata"
}
```

Aufgaben-Metadaten für Amazon ECS-Aufgaben auf EC2 verfügbar

Der Amazon-ECS-Containeragent bietet eine Methode zum Abrufen verschiedener Task-Metadaten und Docker-Statistiken. Dies wird als Endpunkt der Aufgabenmetadaten bezeichnet. Die folgenden Versionen sind verfügbar:

- Endpunkt für Aufgabenmetadaten Version 4: Stellt eine Vielzahl von Metadaten und Docker-Statistiken für Container bereit. Kann auch Daten zur Netzwerkrate bereitstellen. Verfügbar für Amazon-ECS-Aufgaben, die auf Amazon EC2 Linux-Instances gestartet werden, auf denen mindestens Version `1.39.0` des Amazon-ECS-Container-Agenten ausgeführt wird. Für Amazon EC2 Windows-Instances, die `awsvpc`-Netzwerkmodus verwenden, muss der Amazon-ECS-Container-Agent mindestens Version `1.54.0` sein. Weitere Informationen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4.
- Endpunkt für Aufgabenmetadaten Version 3: Stellt eine Vielzahl von Metadaten und Docker-Statistiken für Container bereit. Verfügbar für Amazon-ECS-Aufgaben, die auf Amazon EC2 Linux-

Instances gestartet werden, auf denen mindestens Version `1.21.0` des Amazon-ECS-Container-Agenten ausgeführt wird. Für Amazon EC2 Windows-Instances, die `awsvpc`-Netzwerkmodus verwenden, muss der Amazon-ECS-Container-Agent mindestens Version `1.54.0` sein. Weitere Informationen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 3.

- Endpoint für Aufgabenmetadaten Version 2: Verfügbar für Amazon-ECS-Aufgaben, die auf Amazon EC2 Linux-Instances gestartet werden, auf denen mindestens Version `1.17.0` des Amazon-ECS-Container-Agenten ausgeführt wird. Für Amazon EC2 Windows-Instances, die `awsvpc`-Netzwerkmodus verwenden, muss der Amazon-ECS-Container-Agent mindestens Version `1.54.0` sein. Weitere Informationen finden Sie unter Endpunkt für Amazon ECS-Aufgabenmetadaten, Version 2.

Wenn Ihre Amazon-ECS-Aufgabe auf Amazon EC2 gehostet wird, können Sie auch über den Endpunkt von Instance Metadata Service (IMDS) auf die Metadaten des Aufgaben-Hosts zugreifen. Wenn der folgende Befehl von der Instance aus ausgeführt wird, die die Aufgabe hostet, listet er die ID der Host-Instance auf.

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

Die Informationen, die Sie vom Endpunkt erhalten können, sind in Kategorien unterteilt, wie z. B. *`instance-id`*. Weitere Informationen zu den verschiedenen Kategorien von Host-Instance-Metadaten, die Sie mithilfe des Endpunkts abrufen können, finden Sie unter Kategorien von Instance-Metadaten.

Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4

Der Amazon-ECS-Container-Agent injiziert in jeden Container eine Umgebungsvariable, die als Endpunkt für Task-Metadaten benannt wird, die dem Container verschiedene Task-Metadaten und Docker-Statistiken bereitstellt.

Die Aufgabenmetadaten und Statistiken zur Netzwerkrate werden an CloudWatch Container Insights gesendet und können im eingesehen werden AWS Management Console. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.

> **Note**
>
> Amazon ECS bietet frühere Versionen des Aufgabenmetadaten an. Um in Zukunft keine neuen Endpunktversionen für Aufgabenmetadaten erstellen zu müssen, können zusätzliche

Metadaten zur Ausgabe der Version 4 hinzugefügt werden. Wir werden keine vorhandenen Metadaten entfernen oder Metadatenfeldnamen ändern.

Die Umgebungsvariable wird standardmäßig in die Container von Amazon-ECS-Aufgaben injiziert, die auf Amazon EC2 Linux-Instances gestartet werden, auf denen mindestens Version `1.39.0` des Amazon-ECS-Container-Agenten ausgeführt wird. Für Amazon EC2 Windows-Instances, die `awsvpc`-Netzwerkmodus verwenden, muss der Amazon-ECS-Container-Agent mindestens Version `1.54.0` sein. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.

Note

Sie können Unterstützung für diese Funktion auf Amazon EC2-Instances hinzufügen, indem Sie ältere Versionen der Amazon-ECS-Container-Agenten auf die aktuelle Version aktualisieren. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Pfade für Aufgabenmetadaten-Endpunkt Version 4

Die folgenden Pfade für Aufgaben-Metadaten-Endpunkte sind für Container verfügbar:

`${ECS_CONTAINER_METADATA_URI_V4}`

Dieser Pfad gibt Metadaten für den Container zurück.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Dieser Pfad gibt Metadaten für die Aufgabe zurück, einschließlich einer Liste der Container-IDs und Namen für alle Container, die der Aufgabe zugeordnet sind. Weitere Informationen zu der Antwort für diesen Endpunkt finden Sie unter Amazon ECS-Aufgabenmetadaten V4 JSON-Antwort.

`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`

Dieser Pfad gibt die Metadaten für die Aufgabe zurück, die im `/task`-Endpunkt zusätzlich zu den Task- und Container-Instance-Tags, die mit dem `ListTagsForResource`-API aufgerufen werden können, eingeschlossen sind. Alle Fehler, die beim Abrufen der Tag-Metadaten erhalten werden, sind im `Errors`-Feld in der Antwort enthalten.

> **Note**
>
> Das `Errors`-Feld ist nur in der Antwort für Tasks enthalten, die auf Amazon EC2 Linux-Instances gehostet werden, auf denen mindestens Version `1.50.0` des Container-Agenten ausgeführt wird. Für Amazon EC2 Windows-Instances, die `awsvpc`-Netzwerkmodus verwenden, muss der Amazon-ECS-Container-Agent mindestens Version `1.54.0` sein
> Für diesen Endpunkt ist die `ecs.ListTagsForResource`-Genehmigung erforderlich.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Dieser Pfad gibt Docker-Statistiken für die angegebene Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

Für Amazon-ECS-Aufgaben, die die `awsvpc`- oder `bridge`-Netzwerkmodi nutzen, die auf Amazon EC2 Linux-Instances gehostet werden, auf denen mindestens Version `1.43.0` des Container-Agents ausgeführt werden, werden zusätzliche Netzwerkratenstatistiken in der Antwort enthalten sein. Für alle anderen Aufgaben enthält die Antwort nur die kumulativen Netzwerkstatistiken.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Dieser Pfad gibt Docker-Statistiken für alle der Aufgabe zugeordneten Container zurück. Dies kann von Sidecar-Containern verwendet werden, um Netzwerkmetriken zu extrahieren. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

Für Amazon-ECS-Aufgaben, die die `awsvpc`- oder `bridge`-Netzwerkmodi nutzen, die auf Amazon EC2 Linux-Instances gehostet werden, auf denen mindestens Version `1.43.0` des Container-Agents ausgeführt werden, werden zusätzliche Netzwerkratenstatistiken in der Antwort enthalten sein. Für alle anderen Aufgaben enthält die Antwort nur die kumulativen Netzwerkstatistiken.

Amazon ECS-Aufgabenmetadaten V4 JSON-Antwort

Die folgenden Informationen werden von der JSON-Antwort des Endpunkts für Aufgabenmetadaten (`${ECS_CONTAINER_METADATA_URI_V4}/task`) zurückgegeben. Dazu gehören neben den

Metadaten für jeden Container innerhalb der Aufgabe auch Metadaten, die mit der Aufgabe verknüpft sind.

`Cluster`

Der Amazon-Ressourcenname (ARN) oder kurzer Name des Amazon-ECS-Clusters, zu dem die Aufgabe gehört.

`ServiceName`

Der Name des Dienstes, zu dem die Aufgabe gehört. ServiceName wird für Amazon EC2- und Amazon ECS Anywhere Anywhere-Container-Instances angezeigt, wenn die Aufgabe mit einem Service verknüpft ist.

> **Note**
>
> Diese `ServiceName`-Metadaten sind nur bei Verwendung der Container-Agent-Version `1.63.1` oder höher von Amazon ECS enthalten.

`VPCID`

Die VPC-ID der Amazon-EC2-Container-Instance. Dieses Feld wird nur für Amazon-EC2-Instances angezeigt.

> **Note**
>
> Diese `VPCID`-Metadaten sind nur bei Verwendung der Container-Agent-Version `1.63.1` oder höher von Amazon ECS enthalten.

`TaskARN`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört

`Family`

Die Familie der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`Revision`

Die Revision der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`DesiredStatus`

Der gewünschte Status der Aufgabe von Amazon ECS.

`KnownStatus`

Der bekannte Status der Aufgabe von Amazon ECS.

`Limits`

Die auf der Aufgabenebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`PullStartedAt`

Der Zeitstempel für den Zeitpunkt, an dem der erste Abruf des Container-Image gestartet wurde

`PullStoppedAt`

Der Zeitstempel für den Zeitpunkt, an dem der letzte Abruf des Container-Image abgeschlossen wurde

`AvailabilityZone`

Die Availability Zone, in der sich die Aufgabe befindet.

> **Note**
>
> Die Availability-Zone-Metadaten sind nur für Fargate-Aufgaben verfügbar, die Plattform Version 1.4 oder höher (Linux) oder 1.0.0 (Windows) verwenden.

`LaunchType`

Der Starttyp, den die Aufgabe verwendet. Bei der Verwendung von Clusterkapazitätsanbietern gibt dies an, ob die Aufgabe Fargate oder Amazon EC2-Infrastruktur verwendet.

> **Note**
>
> Diese `LaunchType`-Metadaten sind nur enthalten, wenn Version `1.45.0` oder höher des Amazon-ECs-Linux-Container-Agenten (Linux) oder 1.0.0 oder höher (Windows) verwendet wird.

`Containers`

Eine Liste der Container-Metadaten für jeden Container, der der Aufgabe zugeordnet ist

`DockerId`

Die Docker-ID des Containers

Wenn Sie Fargate verwenden, ist die ID ein 32-stelliger Hex gefolgt von einer 10-stelligen Zahl.

`Name`

Der Name des Containers, wie er in der Aufgabendefinition festgelegt ist

`DockerName`

Der Name des Containers, der vom Docker bereitgestellt wird. Der Amazon-ECS-Containeragent generiert einen eindeutigen Namen für den Container, um Namensüberschneidungen zu vermeiden, wenn mehrere Kopien derselben Aufgabendefinition auf einer einzelnen Instance ausgeführt werden.

`Image`

Das Image für den Container

`ImageID`

Der SHA-256-Digest für das Image

`Ports`

Alle für den Container zugänglichen Ports. Dieser Parameter wird ausgelassen, wenn keine zugänglichen Ports verfügbar sind.

`Labels`

Alle auf den Container angewendeten Kennungen. Dieser Parameter wird ausgelassen, wenn keine angewendeten Kennungen verfügbar sind.

`DesiredStatus`

Der gewünschte Status des Containers von Amazon ECS.

`KnownStatus`

Der bekannte Status des Containers von Amazon ECS.

`ExitCode`

Der Beendigungscode für den Container. Dieser Parameter wird ausgelassen, wenn der Container nicht beendet wurde.

`Limits`

Die auf der Containerebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`CreatedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container erstellt wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht erstellt wurde.

`StartedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container gestartet wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht gestartet wurde.

`FinishedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container angehalten wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht angehalten wurde.

`Type`

Der Typ des Containers. Container, die in der Aufgabendefinition festgelegt sind, haben den Typ `NORMAL`. Sie können andere Containertypen ignorieren, die für die interne Bereitstellung von Aufgabenressourcen durch den Amazon-ECS-Containeragenten verwendet werden.

`LogDriver`

Der Protokolltreiber, den der Container verwendet.

> **Note**
>
> Diese `LogDriver`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`LogOptions`

Die für den Container definierten Protokolltreiber-Optionen.

> **Note**
>
> Diese `LogOptions`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`ContainerARN`

Der vollständige Amazon Resource Name (ARN) des Containers.

> **Note**
>
> Diese `ContainerARN`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`Networks`

Die Netzwerkinformationen für den Container, wie Netzwerkmodus und IP-Adresse. Dieser Parameter wird ausgelassen, wenn keine Netzwerkinformationen definiert sind.

`ExecutionStoppedAt`

Der Zeitstempel des Zeitpunkts, an dem der `DesiredStatus` der Aufgaben zu `STOPPED` gewechselt hat. Dies tritt auf, wenn ein entscheidender Container zu `STOPPED` wechselt.

Beispiele für Amazon ECS-Aufgabenmetadaten v4

Die folgenden Beispiele zeigen Beispielausgaben jeder der Aufgabenmetadaten-Endpunkte.

Beispielantwort für Container-Metadaten

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}` werden nur Metadaten zum Container selbst zurückgegeben. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "DockerId": "ea32192c8553fbff06c9340478a2ff089b2bb5646fb718b4ee206641c9086d66",
    "Name": "curl",
    "DockerName": "ecs-curltest-24-curl-cca48e8dcadd97805600",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
```

```
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/8f03e41243824aea923aca126495f665",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "24"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-02T00:15:07.620912337Z",
    "StartedAt": "2020-10-02T00:15:08.062559351Z",
    "Type": "NORMAL",
    "LogDriver": "awslogs",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/metadata",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/8f03e41243824aea923aca126495f665"
    },
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/0206b271-
b33f-47ab-86c6-a0ba208a70a9",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.100"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0e:9e:32:c7:48:85",
            "IPv4SubnetCIDRBlock": "10.0.2.0/24",
            "PrivateDNSName": "ip-10-0-2-100.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "10.0.2.1/24"
        }
    ]
}
```

Beispiel für eine Aufgabenmetadatenantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/task` werden Metadaten zu der Aufgabe zurückgegeben, zu der der Container gehört, zusätzlich zu den Metadaten für jeden Container innerhalb der Aufgabe. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
```

```
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
```

```
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Beispielaufgabe mit Tags mit Metadatenantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags` werden Metadaten zu der Aufgabe zurückgegeben, einschließlich Aufgabe und Container-Instance-Tags. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "TaskTags": {
```

```
        "tag-use": "task-metadata-endpoint-test"
    },
    "ContainerInstanceTags":{
        "tag_key":"tag_value"
    },
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
```

```
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
```

```
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Beispielaufgabe mit Tags mit einer Fehler-Metadatenantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags` werden Metadaten zu der Aufgabe zurückgegeben, einschließlich Aufgabe und Container-Instance-Tags. Wenn beim Abrufen der Tagging-Daten ein Fehler auftritt, wird der Fehler in der Antwort zurückgegeben. Im Folgenden finden Sie eine Beispielausgabe für den Fall, dass die IAM-Rolle, die der Container-Instance zugeordnet ist, nicht über die `ecs:ListTagsForResource`-Berechtigung verfügt.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "Errors": [
        {
            "ErrorField": "ContainerInstanceTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:container-
instance/default/2dd1b186f39845a584488d2ef155c131",
            "StatusCode": 400,
            "RequestId": "cd597ef0-272b-4643-9bd2-1de469870fa6",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
        },
```

```
        {
            "ErrorField": "TaskTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3",
            "StatusCode": 400,
            "RequestId": "862c5986-6cd2-4aa6-87cc-70be395531e1",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3"
        }
    ],
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
```

```
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
```

```
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Beispielantwort für Container-Statistiken

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/stats` werden Netzwerkmetriken für den Container zurückgegeben. Für Amazon-ECS-Aufgaben, die die `awsvpc`- oder `bridge`-Netzwerkmodi nutzen, die auf Amazon Amazon-EC2-Instances gehostet werden, auf denen mindestens Version `1.43.0` des Container-Agents ausgeführt werden, werden zusätzliche Netzwerkratenstatistiken in der Antwort enthalten sein. Für alle anderen Aufgaben enthält die Antwort nur die kumulativen Netzwerkstatistiken.

Im Folgenden finden Sie eine Beispielausgabe einer Amazon-ECS-Aufgabe auf Amazon EC2, die den `bridge`-Netzwerkmodus nutzt.

```
{
    "read": "2020-10-02T00:51:13.410254284Z",
    "preread": "2020-10-02T00:51:12.406202398Z",
    "pids_stats": {
        "current": 3
    },
    "blkio_stats": {
        "io_service_bytes_recursive": [

        ],
        "io_serviced_recursive": [

        ],
        "io_queue_recursive": [
```

```
        ],
        "io_service_time_recursive": [

        ],
        "io_wait_time_recursive": [

        ],
        "io_merged_recursive": [

        ],
        "io_time_recursive": [

        ],
        "sectors_recursive": [

        ]
    },
    "num_procs": 0,
    "storage_stats": {

    },
    "cpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13939680000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "precpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
```

```
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13937670000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 1806336,
        "max_usage": 6299648,
        "stats": {
            "active_anon": 606208,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 134217728,
            "hierarchical_memsw_limit": 268435456,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 4185,
            "pgmajfault": 0,
            "pgpgin": 2926,
            "pgpgout": 2778,
            "rss": 606208,
            "rss_huge": 0,
            "total_active_anon": 606208,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 4185,
            "total_pgmajfault": 0,
            "total_pgpgin": 2926,
            "total_pgpgout": 2778,
```

```
            "total_rss": 606208,
            "total_rss_huge": 0,
            "total_unevictable": 0,
            "total_writeback": 0,
            "unevictable": 0,
            "writeback": 0
        },
        "limit": 134217728
    },
    "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
    "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
    "networks": {
        "eth0": {
            "rx_bytes": 84,
            "rx_packets": 2,
            "rx_errors": 0,
            "rx_dropped": 0,
            "tx_bytes": 84,
            "tx_packets": 2,
            "tx_errors": 0,
            "tx_dropped": 0
        }
    },
    "network_rate_stats": {
        "rx_bytes_per_sec": 0,
        "tx_bytes_per_sec": 0
    }
}
```

Beispielaufgabe Statistikantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/task/stats` werden Netzwerkmetriken zu der Aufgabe zurückgegeben, zu der der Container gehört. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854": {
        "read": "2020-10-02T00:51:32.51467703Z",
        "preread": "2020-10-02T00:51:31.50860463Z",
        "pids_stats": {
            "current": 1
        },
        "blkio_stats": {
```

```
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13977820000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
```

```
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 532480,
            "max_usage": 6279168,
            "stats": {
                "active_anon": 40960,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 9223372036854771712,
                "hierarchical_memsw_limit": 9223372036854771712,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 2033,
                "pgmajfault": 0,
                "pgpgin": 1734,
                "pgpgout": 1724,
                "rss": 40960,
                "rss_huge": 0,
                "total_active_anon": 40960,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
```

```
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 2033,
                "total_pgmajfault": 0,
                "total_pgpgin": 1734,
                "total_pgpgout": 1724,
                "total_rss": 40960,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 4073377792
        },
        "name": "/ecs-curltest-26-internalecspause-a6bcc3dbadfacfe85300",
        "id": "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    },
    "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af": {
        "read": "2020-10-02T00:51:32.512771349Z",
        "preread": "2020-10-02T00:51:31.510597736Z",
        "pids_stats": {
            "current": 3
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [
```

```
            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 379075681,
                "percpu_usage": [
                    191355275,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13977800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
```

```
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 378825197,
                "percpu_usage": [
                    191104791,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 1814528,
            "max_usage": 6299648,
            "stats": {
                "active_anon": 606208,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 134217728,
                "hierarchical_memsw_limit": 268435456,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 5377,
                "pgmajfault": 0,
                "pgpgin": 3613,
                "pgpgout": 3465,
                "rss": 606208,
                "rss_huge": 0,
                "total_active_anon": 606208,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
```

```
                "total_mapped_file": 0,
                "total_pgfault": 5377,
                "total_pgmajfault": 0,
                "total_pgpgin": 3613,
                "total_pgpgout": 3465,
                "total_rss": 606208,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 134217728
        },
        "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
        "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    }
}
```


Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 3

> **Important**
>
> Der Endpunkt der Version-3-Aufgabenmetadaten wird nicht mehr aktiv verwaltet. Es wird empfohlen, den Endpunkt der Version-4-Aufgabenmetadaten zu aktualisieren, um die

neuesten Metadatenendpunktinformationen zu erhalten Weitere Informationen finden Sie unter the section called “Aufgabenmetadaten-Endpunkt Version 4”.
Wenn Sie Amazon ECS-Aufgaben verwenden, die auf gehostet werden AWS Fargate, finden Sie weitere Informationen unter Task-Metadaten-Endpunkt Version 3 im Amazon Elastic Container Service-Benutzerhandbuch für AWS Fargate.

Ab Version 1.21.0 des Amazon-ECS-Containeragenten führt der Agent eine Umgebungsvariable namens `ECS_CONTAINER_METADATA_URI` in jeden Container einer Aufgabe ein. Wenn Sie den Endpunkt der Aufgaben-Metadaten Version 3 abfragen, stehen für die Aufgaben verschiedene Aufgaben-Metadaten und Docker Statistiken zur Verfügung. Bei Aufgaben, die den Netzwerkmodus `bridge` verwenden, stehen Netzwerkmetriken zur Verfügung, wenn die `/stats`-Endpunkte abgefragt werden.

Die Funktionalität vom Feature Aufgabenmetadaten-Endpunkt Version 3 ist standardmäßig für Aufgaben aktiviert, die den Starttyp Fargate auf der Plattform-Version v1.3.0 oder höher verwenden. Sie ist zudem für Aufgaben aktiviert, die den Starttyp EC2 verwenden und auf einer Amazon EC2 Linux-Infrastruktur gestartet werden, auf der mindestens Version 1.21.0 des Amazon-ECS-Container-Agenten oder auf Amazon-EC2-Windows-Infrastruktur die mindestens auf Version `1.54.0` des Amazon-ECS-Container-Agenten ausgeführt wird und den `awsvpc`-Netzwerkmodus nutzt. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.

Sie können Unterstützung für dieses Feature auf älteren Container-Instances hinzufügen, indem Sie den Agenten auf die aktuelle Version aktualisieren. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Important

Für Aufgaben, die den Starttyp Fargate und die Plattform-Versionen vor v1.3.0 verwenden, wird der Endpunkt von Aufgaben-Metadaten Version 2 unterstützt. Weitere Informationen finden Sie unter Endpunkt für Amazon ECS-Aufgabenmetadaten, Version 2.

Pfade zum Endpunkt für Aufgaben-Metadaten, Version 3

Die folgenden Aufgaben-Metadaten-Endpunkte sind für Container verfügbar:

`${ECS_CONTAINER_METADATA_URI}`

Dieser Pfad gibt JSON-Metadaten für den Container zurück.

`${ECS_CONTAINER_METADATA_URI}/task`

Dieser Pfad gibt JSON-Metadaten für die Aufgabe zurück, einschließlich einer Liste der Container-IDs und Namen für alle Container, die der Aufgabe zugeordnet sind. Weitere Informationen zu der Antwort für diesen Endpunkt finden Sie unter JSON-Antwort mit Amazon ECS-Aufgabenmetadaten v3.

`${ECS_CONTAINER_METADATA_URI}/taskWithTags`

Dieser Pfad gibt die Metadaten für die Aufgabe zurück, die im `/task`-Endpunkt zusätzlich zu den Task- und Container-Instance-Tags, die mit dem `ListTagsForResource`-API aufgerufen werden können, eingeschlossen sind.

`${ECS_CONTAINER_METADATA_URI}/stats`

Dieser Pfad gibt JSON-Docker-Statistiken für die angegebene Docker-Container-ID zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Dieser Pfad gibt JSON-Docker-Statistiken für alle der Aufgabe zugeordneten Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

JSON-Antwort mit Amazon ECS-Aufgabenmetadaten v3

Die folgenden Informationen werden von der JSON-Antwort des Endpunkts für Aufgabenmetadaten (`${ECS_CONTAINER_METADATA_URI}/task`) zurückgegeben.

`Cluster`

Der Amazon-Ressourcenname (ARN) oder kurzer Name des Amazon-ECS-Clusters, zu dem die Aufgabe gehört.

`TaskARN`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört

`Family`

Die Familie der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`Revision`

Die Revision der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`DesiredStatus`

Der gewünschte Status der Aufgabe von Amazon ECS.

`KnownStatus`

Der bekannte Status der Aufgabe von Amazon ECS.

`Limits`

Die auf der Aufgabenebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`PullStartedAt`

Der Zeitstempel für den Zeitpunkt, an dem der erste Abruf des Container-Image gestartet wurde

`PullStoppedAt`

Der Zeitstempel für den Zeitpunkt, an dem der letzte Abruf des Container-Image abgeschlossen wurde

`AvailabilityZone`

Die Availability Zone, in der sich die Aufgabe befindet.

> **Note**
>
> Die Availability-Zone-Metadaten sind nur für Fargate-Aufgaben verfügbar, die Plattform Version 1.4 oder höher (Linux) oder 1.0.0 oder höher (Windows) verwenden.

`Containers`

Eine Liste der Container-Metadaten für jeden Container, der der Aufgabe zugeordnet ist

`DockerId`

Die Docker-ID des Containers

`Name`

Der Name des Containers, wie er in der Aufgabendefinition festgelegt ist

`DockerName`

Der Name des Containers, der vom Docker bereitgestellt wird. Der Amazon-ECS-Containeragent generiert einen eindeutigen Namen für den Container, um Namensüberschneidungen zu vermeiden, wenn mehrere Kopien derselben Aufgabendefinition auf einer einzelnen Instance ausgeführt werden.

`Image`

Das Image für den Container

`ImageID`

Der SHA-256-Digest für das Image

`Ports`

Alle für den Container zugänglichen Ports. Dieser Parameter wird ausgelassen, wenn keine zugänglichen Ports verfügbar sind.

`Labels`

Alle auf den Container angewendeten Kennungen. Dieser Parameter wird ausgelassen, wenn keine angewendeten Kennungen verfügbar sind.

`DesiredStatus`

Der gewünschte Status des Containers von Amazon ECS.

`KnownStatus`

Der bekannte Status des Containers von Amazon ECS.

`ExitCode`

Der Beendigungscode für den Container. Dieser Parameter wird ausgelassen, wenn der Container nicht beendet wurde.

`Limits`

Die auf der Containerebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`CreatedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container erstellt wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht erstellt wurde.

`StartedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container gestartet wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht gestartet wurde.

`FinishedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container angehalten wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht angehalten wurde.

`Type`

Der Typ des Containers. Container, die in der Aufgabendefinition festgelegt sind, haben den Typ `NORMAL`. Sie können andere Containertypen ignorieren, die für die interne Bereitstellung von Aufgabenressourcen durch den Amazon-ECS-Containeragenten verwendet werden.

`Networks`

Die Netzwerkinformationen für den Container, wie Netzwerkmodus und IP-Adresse. Dieser Parameter wird ausgelassen, wenn keine Netzwerkinformationen definiert sind.

`ClockDrift`

Die Informationen über die Abweichung zwischen der Referenzzeit und der Systemzeit. Dies gilt für das Linux-Betriebssystem. Diese Funktion verwendet den Amazon Time Sync Service, um die Genauigkeit der Uhr zu messen und den Zeitfehler für Container bereitzustellen. Weitere Informationen finden Sie unter Zeit für Ihre Linux-Instance festlegen im Amazon EC2 EC2-Benutzerhandbuch für Linux-Instances.

`ReferenceTime`

Die Grundlage der Taktgenauigkeit. Amazon ECS verwendet beispielsweise den globalen Standard Coordinated Universal Time (UTC) über NTP, z. B. `2021-09-07T16:57:44Z`.

`ClockErrorBound`

Das Maß des Taktfehlers, definiert als Abweichung zu UTC. Dieser Fehler ist der Unterschied in Millisekunden zwischen der Referenzzeit und der Systemzeit.

`ClockSynchronizationStatus`

Gibt an, ob der letzte Synchronisierungsversuch zwischen der Systemzeit und der Referenzzeit erfolgreich war.

Die gültigen Werte sind `SYNCHRONIZED` und `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Der Zeitstempel des Zeitpunkts, an dem der `DesiredStatus` der Aufgaben zu `STOPPED` gewechselt hat. Dies tritt auf, wenn ein entscheidender Container zu `STOPPED` wechselt.

Beispiele für Amazon ECS-Aufgabenmetadaten v3

Die folgenden Beispiele zeigen Beispielausgaben der Aufgabenmetadaten-Endpunkte.

Beispielantwort für Container-Metadaten

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI}` werden nur Metadaten zum Container selbst zurückgegeben. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
    "Name": "nginx-curl",
    "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
    "Image": "nrdlngr/nginx-curl",
    "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
```

```
    "Limits": {
        "CPU": 512,
        "Memory": 512
    },
    "CreatedAt": "2018-02-01T20:55:10.554941919Z",
    "StartedAt": "2018-02-01T20:55:11.064236631Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.106"
            ]
        }
    ]
}
```

Beispiel für eine Aufgabenmetadatenantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI}/task` werden Metadaten zu der Aufgabe zurückgegeben, zu der der Container gehört. Im Folgenden finden Sie eine Beispielausgabe.

Die folgende JSON-Antwort ist für eine Aufgabe mit einem Container.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
```

```
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
```

```
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Endpunkt für Amazon ECS-Aufgabenmetadaten, Version 2

> **Important**
>
> Der Endpunkt der Aufgabenmetadaten Version 2 wird nicht mehr aktiv verwaltet. Es wird empfohlen, den Endpunkt der Version-4-Aufgabenmetadaten zu aktualisieren, um die neuesten Metadatenendpunktinformationen zu erhalten Weitere Informationen finden Sie unter the section called “Aufgabenmetadaten-Endpunkt Version 4”.

Ab Version 1.17.0 des Amazon-ECS-Containeragenten sind für Aufgaben, die den `awsvpc`-Netzwerkmodus für einen HTTP-Endpunkt verwenden, der vom Amazon-ECS-Containeragenten bereitgestellt wird, verschiedene Aufgabenmetadaten und Docker-Statistiken verfügbar.

Alle Container von Aufgaben, die mit dem Netzwerkmodus `awsvpc` gestartet werden, erhalten eine lokale IPv4-Adresse innerhalb eines vordefinierten link-local-Adressbereichs. Wenn ein Container den Metadatenendpunkt abfragt, kann der Amazon-ECS-Containeragent anhand der eindeutigen IP-Adresse die Aufgabe bestimmen, zu der der Container gehört, und gibt die Metadaten und Statistiken für diese Aufgabe zurück.

Aktivieren von Aufgabenmetadaten

Das Feature Aufgaben-Metadaten-Version 2 ist standardmäßig für Folgendes aktiviert:

- Aufgaben mit dem Starttyp Fargate, die die Plattform-Version v1.1.0 oder höher verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.
- Aufgaben, die den EC2-Starttyp und auch den `awsvpc`-Netzwerkmodus verwenden und in der Amazon EC2 Linux-Infrastruktur ab Version 1.17.0 des Amazon-ECS-Containeragenten ausgeführt werden, oder in der Amazon EC2-Windows-Infrastruktur ab Version `1.54.0` des Amazon-ECS-Container-Agenten ausgeführt werden. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.

Sie können Unterstützung für dieses Feature auf älteren Container-Instances hinzufügen, indem Sie den Agenten auf die aktuelle Version aktualisieren. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Endpunktpfade für Aufgabenmetadaten

Die folgenden API-Endpunkte sind für Container verfügbar:

`169.254.170.2/v2/metadata`

Dieser Endpunkt gibt Metadaten-JSON für die Aufgabe zurück, einschließlich einer Liste der Container-IDs und Namen für alle Container, die der Aufgabe zugeordnet sind. Weitere Informationen zu der Antwort für diesen Endpunkt finden Sie unter Aufgabenmetadaten-JSON-Antwort.

`169.254.170.2/v2/metadata/<container-id>`

Dieser Endpunkt gibt Metadaten-JSON für die angegebene Docker-Container-ID zurück.

`169.254.170.2/v2/metadata/taskWithTags`

Dieser Pfad gibt die Metadaten für die Aufgabe zurück, die im `/task`-Endpunkt zusätzlich zu den Task- und Container-Instance-Tags, die mit dem `ListTagsForResource`-API aufgerufen werden können, eingeschlossen sind.

`169.254.170.2/v2/stats`

Dieser Endpunkt gibt Docker-Statistik-JSON für alle der Aufgabe zugeordneten Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

`169.254.170.2/v2/stats/<container-id>`

Dieser Endpunkt gibt Docker-Statistik-JSON für die angegebene Docker-Container-ID zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

Aufgabenmetadaten-JSON-Antwort

Die folgenden Informationen werden von der JSON-Antwort des Endpunkts für Aufgabenmetadaten (`169.254.170.2/v2/metadata`) zurückgegeben.

`Cluster`

Der Amazon-Ressourcenname (ARN) oder kurzer Name des Amazon-ECS-Clusters, zu dem die Aufgabe gehört.

`TaskARN`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört

`Family`

Die Familie der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`Revision`

Die Revision der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`DesiredStatus`

Der gewünschte Status der Aufgabe von Amazon ECS.

`KnownStatus`

Der bekannte Status der Aufgabe von Amazon ECS.

`Limits`

Die auf der Aufgabenebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`PullStartedAt`

Der Zeitstempel für den Zeitpunkt, an dem der erste Abruf des Container-Image gestartet wurde

`PullStoppedAt`

Der Zeitstempel für den Zeitpunkt, an dem der letzte Abruf des Container-Image abgeschlossen wurde

`AvailabilityZone`

Die Availability Zone, in der sich die Aufgabe befindet.

> **Note**
>
> Die Availability-Zone-Metadaten sind nur für Fargate-Aufgaben verfügbar, die Plattform Version 1.4 oder höher (Linux) oder 1.0.0 oder höher (Windows) verwenden.

`Containers`

Eine Liste der Container-Metadaten für jeden Container, der der Aufgabe zugeordnet ist

`DockerId`

Die Docker-ID des Containers

`Name`

Der Name des Containers, wie er in der Aufgabendefinition festgelegt ist

`DockerName`

Der Name des Containers, der vom Docker bereitgestellt wird. Der Amazon-ECS-Containeragent generiert einen eindeutigen Namen für den Container, um Namensüberschneidungen zu vermeiden, wenn mehrere Kopien derselben Aufgabendefinition auf einer einzelnen Instance ausgeführt werden.

`Image`

Das Image für den Container

`ImageID`

Der SHA-256-Digest für das Image

`Ports`

Alle für den Container zugänglichen Ports. Dieser Parameter wird ausgelassen, wenn keine zugänglichen Ports verfügbar sind.

`Labels`

Alle auf den Container angewendeten Kennungen. Dieser Parameter wird ausgelassen, wenn keine angewendeten Kennungen verfügbar sind.

`DesiredStatus`

Der gewünschte Status des Containers von Amazon ECS.

`KnownStatus`

Der bekannte Status des Containers von Amazon ECS.

`ExitCode`

Der Beendigungscode für den Container. Dieser Parameter wird ausgelassen, wenn der Container nicht beendet wurde.

`Limits`

Die auf der Containerebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`CreatedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container erstellt wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht erstellt wurde.

`StartedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container gestartet wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht gestartet wurde.

`FinishedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container angehalten wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht angehalten wurde.

`Type`

Der Typ des Containers. Container, die in der Aufgabendefinition festgelegt sind, haben den Typ `NORMAL`. Sie können andere Containertypen ignorieren, die für die interne Bereitstellung von Aufgabenressourcen durch den Amazon-ECS-Containeragenten verwendet werden.

`Networks`

Die Netzwerkinformationen für den Container, wie Netzwerkmodus und IP-Adresse. Dieser Parameter wird ausgelassen, wenn keine Netzwerkinformationen definiert sind.

`ClockDrift`

Die Informationen über die Abweichung zwischen der Referenzzeit und der Systemzeit. Dies gilt für das Linux-Betriebssystem. Diese Funktion verwendet den Amazon Time Sync Service, um die Genauigkeit der Uhr zu messen und den Zeitfehler für Container bereitzustellen. Weitere Informationen finden Sie unter Zeit für Ihre Linux-Instance festlegen im Amazon EC2 EC2-Benutzerhandbuch für Linux-Instances.

`ReferenceTime`

Die Grundlage der Taktgenauigkeit. Amazon ECS verwendet beispielsweise den globalen Standard Coordinated Universal Time (UTC) über NTP, z. B. `2021-09-07T16:57:44Z`.

`ClockErrorBound`

Das Maß des Taktfehlers, definiert als Abweichung zu UTC. Dieser Fehler ist der Unterschied in Millisekunden zwischen der Referenzzeit und der Systemzeit.

`ClockSynchronizationStatus`

Gibt an, ob der letzte Synchronisierungsversuch zwischen der Systemzeit und der Referenzzeit erfolgreich war.

Die gültigen Werte sind `SYNCHRONIZED` und `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Der Zeitstempel des Zeitpunkts, an dem der `DesiredStatus` der Aufgaben zu `STOPPED` gewechselt hat. Dies tritt auf, wenn ein entscheidender Container zu `STOPPED` wechselt.

Beispiel für eine Aufgabenmetadatenantwort

Die folgende JSON-Antwort ist für eine Aufgabe mit einem Container.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
```

```
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
```

```
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Amazon ECS-Aufgabenmetadaten für Aufgaben auf Fargate verfügbar

Amazon ECS auf Fargate bietet eine Methode zum Abrufen verschiedener Metadaten, Netzwerkmetriken und Docker-Statistiken zu Ihren Containern und den Aufgaben, von denen sie ein Teil sind. Dies wird als Endpunkt der Aufgabenmetadaten bezeichnet. Die folgenden Endpunktversionen für Aufgabenmetadaten sind für Amazon ECS auf Fargate-Aufgaben verfügbar:

- Aufgabenmetadaten-Endpunkt Version 4: Verfügbar für Aufgaben, die die Plattformversion 1.4.0 oder höher nutzen.
- Aufgabenmetadaten-Endpunkt Version 3: Verfügbar für Aufgaben, die die Plattformversion 1.1.0 oder höher nutzen.

Alle Container von Aufgaben, die mit dem Netzwerkmodus `awsvpc` gestartet werden, erhalten eine lokale IPv4-Adresse innerhalb eines vordefinierten link-local-Adressbereichs. Wenn ein Container den Metadatenendpunkt abfragt, kann der -Containeragent anhand der eindeutigen IP-Adresse die Aufgabe bestimmen, zu der der Container gehört, und gibt die Metadaten und Statistiken für diese Aufgabe zurück.

Themen

Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4, für Aufgaben auf Fargate

> **Important**
>
> Wenn Sie Amazon ECS-Aufgaben verwenden, die auf Amazon EC2 EC2-Instances gehostet werden, finden Sie weitere Informationen unter Amazon ECS-Aufgabenmetadaten-Endpunkt.

Ab Fargate-Plattformversion `1.4.0` wird in jeden Container einer Aufgabe eine Umgebungsvariable mit dem Namen `ECS_CONTAINER_METADATA_URI_V4` eingeschleust. Wenn Sie den Aufgabenmetadaten-Endpunkt Version 4 abfragen, stehen für die Aufgaben verschiedene Aufgabenmetadaten und Docker-Statistiken zur Verfügung.

Der Aufgabenmetadaten-Endpunkt Version 4 funktioniert wie der Endpunkt der Version 3, stellt jedoch zusätzliche Netzwerkmetadaten für Ihre Container und Aufgaben bereit. Zusätzliche Netzwerkmetriken sind ebenfalls verfügbar, wenn die `/stats`-Endpunkte abgefragt werden.

Der Endpunkt für Aufgabenmetadaten ist standardmäßig für alle Amazon ECS-Aufgaben aktiviert AWS Fargate , die auf dieser Nutzungsplattformversion `1.4.0` oder höher ausgeführt werden.

> **Note**
>
> Um in Zukunft keine neuen Endpunktversionen für Aufgabenmetadaten erstellen zu müssen, können zusätzliche Metadaten zur Ausgabe der Version 4 hinzugefügt werden. Wir werden keine vorhandenen Metadaten entfernen oder Metadatenfeldnamen ändern.

Pfade für Fagate-Aufgabenmetadaten-Endpunkt Version 4

Die folgenden Aufgaben-Metadaten-Endpunkte sind für Container verfügbar:

`${ECS_CONTAINER_METADATA_URI_V4}`

Dieser Pfad gibt Metadaten für den Container zurück.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Dieser Pfad gibt Metadaten für die Aufgabe zurück, einschließlich einer Liste der Container-IDs und Namen für alle Container, die der Aufgabe zugeordnet sind. Weitere Informationen zu der Antwort für diesen Endpunkt finden Sie unter Amazon ECS-Aufgabenmetadaten v4 JSON-Antwort für Aufgaben auf Fargate.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Dieser Pfad gibt Docker-Statistiken für den Docker-Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

> **Note**
>
> Bei aktivierten Amazon ECS-Aufgaben AWS Fargate muss der Container ~1 Sekunde lang laufen, bevor die Container-Statistiken zurückgegeben werden.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Dieser Pfad gibt Docker-Statistiken für alle der Aufgabe zugeordneten Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

> **Note**
>
> Bei aktivierten Amazon ECS-Aufgaben AWS Fargate muss der Container ~1 Sekunde lang laufen, bevor die Container-Statistiken zurückgegeben werden.

Amazon ECS-Aufgabenmetadaten v4 JSON-Antwort für Aufgaben auf Fargate

Die folgenden Metadaten werden in der JSON-Antwort des Endpunkts für Aufgabenmetadaten (`${ECS_CONTAINER_METADATA_URI_V4}/task`) zurückgegeben.

`Cluster`

Der Amazon-Ressourcenname (ARN) oder kurzer Name des Amazon ECS-Clusters, zu dem die Aufgabe gehört.

`VPCID`

Die VPC-ID der Amazon-EC2-Container-Instance. Dieses Feld wird nur für Amazon-EC2-Instances angezeigt.

> **Note**
>
> Diese `VPCID`-Metadaten sind nur bei Verwendung der Container-Agent-Version `1.63.1` oder höher von Amazon ECS enthalten.

`TaskARN`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört

`Family`

Die Familie der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`Revision`

Die Revision der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`DesiredStatus`

Der gewünschte Status der Aufgabe von Amazon ECS.

`KnownStatus`

Der bekannte Status der Aufgabe von Amazon ECS.

`Limits`

Die auf den Aufgabenebenen festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`PullStartedAt`

Der Zeitstempel für den Zeitpunkt, an dem der erste Abruf des Container-Image gestartet wurde

`PullStoppedAt`

Der Zeitstempel für den Zeitpunkt, an dem der letzte Abruf des Container-Image abgeschlossen wurde

`AvailabilityZone`

Die Availability Zone, in der sich die Aufgabe befindet.

> **Note**
>
> Die Availability-Zone-Metadaten sind nur für Fargate-Aufgaben verfügbar, die Plattform Version 1.4 oder höher (Linux) oder 1.0.0 (Windows) verwenden.

`LaunchType`

Der Starttyp, den die Aufgabe verwendet. Bei der Verwendung von Clusterkapazitätsanbietern gibt dies an, ob die Aufgabe Fargate oder Amazon EC2-Infrastruktur verwendet.

> **Note**
>
> Diese `LaunchType`-Metadaten sind nur enthalten, wenn Version `1.45.0` oder höher des Amazon-ECs-Linux-Container-Agenten (Linux) oder 1.0.0 oder höher (Windows) verwendet wird.

`EphemeralStorageMetrics`

Die reservierte Größe und aktuelle Nutzung des flüchtigen Speichers dieser Aufgabe.

> **Note**
>
> Fargate reserviert Speicherplatz auf der Festplatte. Er wird nur von Fargate verwendet. Ihnen entstehen dafür keine Kosten. Es wird in diesen Metriken nicht angezeigt. Sie können diesen zusätzlichen Speicherplatz jedoch in anderen Tools sehen, wie z. B. `df`.

`Utilized`

Die aktuelle flüchtige Speichernutzung (in MiB) dieser Aufgabe.

`Reserved`

Der reservierte flüchtige Speicher (in MiB) dieser Aufgabe. Die Größe des flüchtigen Speichers kann in einer ausgeführten Aufgabe nicht geändert werden. Sie können das `ephermalStorage`-Objekt in Ihrer Aufgabendefinition angeben, um die Menge des flüchtigen Speichers zu ändern. Der `ephermalStorage` wird in GB angegeben, nicht in MB. Der `ephermalStorage` und die `EphemeralStorageMetrics` sind nur auf Fargate-Linux-Plattform-Version 1.4.0 oder höher verfügbar.

`Containers`

Eine Liste der Container-Metadaten für jeden Container, der der Aufgabe zugeordnet ist

`DockerId`

Die Docker-ID des Containers

Wenn Sie Fargate verwenden, ist die ID ein 32-stelliger Hex gefolgt von einer 10-stelligen Zahl.

`Name`

Der Name des Containers, wie er in der Aufgabendefinition festgelegt ist

`DockerName`

Der Name des Containers, der vom Docker bereitgestellt wird. Der Amazon-ECS-Containeragent generiert einen eindeutigen Namen für den Container, um Namensüberschneidungen zu vermeiden, wenn mehrere Kopien derselben Aufgabendefinition auf einer einzelnen Instance ausgeführt werden.

`Image`

Das Image für den Container

`ImageID`

Der SHA-256-Digest für das Image

`Ports`

Alle für den Container zugänglichen Ports. Dieser Parameter wird ausgelassen, wenn keine zugänglichen Ports verfügbar sind.

`Labels`

Alle auf den Container angewendeten Kennungen. Dieser Parameter wird ausgelassen, wenn keine angewendeten Kennungen verfügbar sind.

`DesiredStatus`

Der gewünschte Status des Containers von Amazon ECS.

`KnownStatus`

Der bekannte Status des Containers von Amazon ECS.

`ExitCode`

Der Beendigungscode für den Container. Dieser Parameter wird ausgelassen, wenn der Container nicht beendet wurde.

`Limits`

Die auf den Containerebenen festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`CreatedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container erstellt wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht erstellt wurde.

`StartedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container gestartet wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht gestartet wurde.

`FinishedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container angehalten wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht angehalten wurde.

`Type`

Der Typ des Containers. Container, die in der Aufgabendefinition festgelegt sind, haben den Typ `NORMAL`. Sie können andere Containertypen ignorieren, die für die interne Bereitstellung von Aufgabenressourcen durch den Amazon-ECS-Containeragenten verwendet werden.

`LogDriver`

Der Protokolltreiber, den der Container verwendet.

> **Note**
>
> Diese `LogDriver`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`LogOptions`

Die für den Container definierten Protokolltreiber-Optionen.

> **Note**
>
> Diese `LogOptions`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`ContainerARN`

Der vollständige Amazon Resource Name (ARN) des Containers.

> **Note**
>
> Diese `ContainerARN`-Metadaten sind nur bei Verwendung der Amazon ECS Linux-Container-Agent-Version `1.45.0` oder höher enthalten.

`Networks`

Die Netzwerkinformationen für den Container, wie Netzwerkmodus und IP-Adresse. Dieser Parameter wird ausgelassen, wenn keine Netzwerkinformationen definiert sind.

`Snapshotter`

snapshotter, der von containerd zum Herunterladen dieses Container-Images verwendet wurde. Gültige Werte sind `overlayfs`, was die Standardeinstellung ist, und `soci`, die beim verzögerten Laden mit einem SOCI-Index verwendet werden. Dieser Parameter ist nur für Aufgaben verfügbar, die auf Linux-Plattformversion `1.4.0` ausgeführt werden.

`ClockDrift`

Die Informationen über die Abweichung zwischen der Referenzzeit und der Systemzeit. Diese Funktion verwendet den Amazon Time Sync Service, um die Genauigkeit der Uhr zu messen und

den Zeitfehler für Container bereitzustellen. Weitere Informationen finden Sie unter Zeit für Ihre Linux-Instance festlegen im Amazon EC2 EC2-Benutzerhandbuch für Linux-Instances.

`ReferenceTime`

Die Grundlage der Taktgenauigkeit. Amazon ECS verwendet beispielsweise den globalen Standard Coordinated Universal Time (UTC) über NTP, z. B. `2021-09-07T16:57:44Z`.

`ClockErrorBound`

Das Maß des Taktfehlers, definiert als Abweichung zu UTC. Dieser Fehler ist der Unterschied in Millisekunden zwischen der Referenzzeit und der Systemzeit.

`ClockSynchronizationStatus`

Gibt an, ob der letzte Synchronisierungsversuch zwischen der Systemzeit und der Referenzzeit erfolgreich war.

Die gültigen Werte sind `SYNCHRONIZED` und `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Der Zeitstempel des Zeitpunkts, an dem der `DesiredStatus` der Aufgaben zu `STOPPED` gewechselt hat. Dies tritt auf, wenn ein entscheidender Container zu `STOPPED` wechselt.

Beispiele für Amazon ECS-Aufgabenmetadaten v4 für Aufgaben auf Fargate

Die folgenden Beispiele zeigen Beispielausgaben der Aufgabenmetadaten-Endpunkte für Amazon ECS-Aufgaben, die auf AWS Fargate ausgeführt werden.

Vom Container aus können Sie curl gefolgt vom Endpunkt der Aufgaben-Metadaten verwenden, um den Endpunkt abzufragen, z. B. `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`

Beispielantwort für Container-Metadaten

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}` werden nur Metadaten zum Container selbst zurückgegeben. Im Folgenden finden Sie eine Beispielausgabe.

```
{
    "DockerId": "cd189a933e5849daa93386466019ab50-2495160603",
    "Name": "curl",
    "DockerName": "curl",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:25f3695bedfb454a50f12d127839a68ad3caf91e451c1da073db34c542c4d2cb",
```

```
    "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/default/
cd189a933e5849daa93386466019ab50",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "2"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-08T20:09:11.44527186Z",
    "StartedAt": "2020-10-08T20:09:11.44527186Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "192.0.2.3"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0a:de:f6:10:51:e5",
            "IPv4SubnetCIDRBlock": "192.0.2.0/24",
            "DomainNameServers": [
                "192.0.2.2"
            ],
            "DomainNameSearchList": [
                "us-west-2.compute.internal"
            ],
            "PrivateDNSName": "ip-10-0-0-222.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "192.0.2.0/24"
        }
    ],
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/05966557-
f16c-49cb-9352-24b3a0dcd0e1",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/containerlogs",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/cd189a933e5849daa93386466019ab50"
```

```
    },
    "LogDriver": "awslogs",
    "Snapshotter": "overlayfs"
}
```

Beispiele für Amazon ECS-Aufgabenmetadaten v4 für Aufgaben auf Fargate

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/task` werden Metadaten zu der Aufgabe zurückgegeben, zu der der Container gehört. Im Folgenden finden Sie eine Beispielausgabe.

```
{
  "Cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/clusterName",
  "TaskARN": "arn:aws:ecs:us-east-1:123456789012:task/MyEmptyCluster/
bfa2636268144d039771334145e490c5",
  "Family": "sample-fargate",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": {
    "CPU": 0.25,
    "Memory": 512
  },
  "PullStartedAt": "2023-07-21T15:45:33.532811081Z",
  "PullStoppedAt": "2023-07-21T15:45:38.541068435Z",
  "AvailabilityZone": "us-east-1d",
  "Containers": [
    {
      "DockerId": "bfa2636268144d039771334145e490c5-1117626119",
      "Name": "curl-image",
      "DockerName": "curl-image",
      "Image": "curlimages/curl",
      "ImageID":
 "sha256:daf3f46a2639c1613b25e85c9ee4193af8a1d538f92483d67f9a3d7f21721827",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "curl-image",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
```

```
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 128 },
      "CreatedAt": "2023-07-21T15:45:44.91368314Z",
      "StartedAt": "2023-07-21T15:45:44.91368314Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/da6cccf7-1178-400c-afdf-7536173ee209",
      "Snapshotter": "overlayfs"
    },
    {
      "DockerId": "bfa2636268144d039771334145e490c5-3681984407",
      "Name": "fargate-app",
      "DockerName": "fargate-app",
      "Image": "public.ecr.aws/docker/library/httpd:latest",
      "ImageID":
 "sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "fargate-app",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2023-07-21T15:45:44.954460255Z",
      "StartedAt": "2023-07-21T15:45:44.954460255Z",
```

```
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/f65b461d-aa09-4acb-a579-9785c0530cbc",
      "Snapshotter": "overlayfs"
    }
  ],
  "LaunchType": "FARGATE",
  "ClockDrift": {
    "ClockErrorBound": 0.446931,
    "ReferenceTimestamp": "2023-07-21T16:09:17Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "EphemeralStorageMetrics": {
    "Utilized": 261,
    "Reserved": 20496
  }
}
```

Beispielaufgabe Statistikantwort

Beim Abfragen des Endpunkts `${ECS_CONTAINER_METADATA_URI_V4}/task/stats` werden Netzwerkmetriken zu der Aufgabe zurückgegeben, zu der der Container gehört. Im Folgenden finden Sie eine Beispielausgabe.

```
{
  "3d1f891cded94dc795608466cce8ddcf-464223573": {
    "read": "2020-10-08T21:24:44.938937019Z",
    "preread": "2020-10-08T21:24:34.938633969Z",
    "pids_stats": {},
    "blkio_stats": {
```

```
      "io_service_bytes_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 638976
        }
      ],
      "io_serviced_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
```

```
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 12
        }
      ],
      "io_queue_recursive": [],
      "io_service_time_recursive": [],
      "io_wait_time_recursive": [],
      "io_merged_recursive": [],
      "io_time_recursive": [],
      "sectors_recursive": []
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
        "total_usage": 1137691504,
        "percpu_usage": [
          696479228,
          441212276,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
```

```
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9393210000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 1136624601,
        "percpu_usage": [
          695639662,
          440984939,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9373330000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
```

```
        "throttled_time": 0
      }
    },
    "memory_stats": {
      "usage": 6504448,
      "max_usage": 8458240,
      "stats": {
        "active_anon": 1675264,
        "active_file": 557056,
        "cache": 651264,
        "dirty": 0,
        "hierarchical_memory_limit": 536870912,
        "hierarchical_memsw_limit": 9223372036854772000,
        "inactive_anon": 0,
        "inactive_file": 3088384,
        "mapped_file": 430080,
        "pgfault": 11034,
        "pgmajfault": 5,
        "pgpgin": 8436,
        "pgpgout": 7137,
        "rss": 4669440,
        "rss_huge": 0,
        "total_active_anon": 1675264,
        "total_active_file": 557056,
        "total_cache": 651264,
        "total_dirty": 0,
        "total_inactive_anon": 0,
        "total_inactive_file": 3088384,
        "total_mapped_file": 430080,
        "total_pgfault": 11034,
        "total_pgmajfault": 5,
        "total_pgpgin": 8436,
        "total_pgpgout": 7137,
        "total_rss": 4669440,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 0,
        "unevictable": 0,
        "writeback": 0
      },
      "limit": 9223372036854772000
    },
    "name": "curltest",
    "id": "3d1f891cded94dc795608466cce8ddcf-464223573",
```

```
    "networks": {
      "eth1": {
        "rx_bytes": 2398415937,
        "rx_packets": 1898631,
        "rx_errors": 0,
        "rx_dropped": 0,
        "tx_bytes": 1259037719,
        "tx_packets": 428002,
        "tx_errors": 0,
        "tx_dropped": 0
      }
    },
    "network_rate_stats": {
      "rx_bytes_per_sec": 43.298687872232854,
      "tx_bytes_per_sec": 215.39347269466413
    }
  }
}
```

Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 3, für Aufgaben auf Fargate

> **Important**
>
> Der Endpunkt der Version-3-Aufgabenmetadaten wird nicht mehr aktiv verwaltet. Es wird empfohlen, den Endpunkt der Version-4-Aufgabenmetadaten zu aktualisieren, um die neuesten Metadatenendpunktinformationen zu erhalten Weitere Informationen finden Sie unter the section called “Aufgabenmetaden-Endpunkt Version 4 für Aufgaben auf Fargate”.

Ab Fargate-Plattformversion `1.1.0` wird in jeden Container einer Aufgabe eine Umgebungsvariable mit dem Namen `ECS_CONTAINER_METADATA_URI` eingeschleust. Wenn Sie den Endpunkt der Aufgaben-Metadaten Version 3 abfragen, stehen für die Aufgaben verschiedene Aufgaben-Metadaten und Docker Statistiken zur Verfügung.

Das Feature Aufgabenmetadaten-Endpunkt ist standardmäßig für alle Amazon ECS-Aufgaben aktiviert, die auf Fargate gehostet werden und die Plattformversion `1.1.0` oder höher verwenden. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.

Pfade für Aufgabenmetaden-Endpunkt für Aufgaben auf Fargate

Die folgenden API-Endpunkte sind für Container verfügbar:

`${ECS_CONTAINER_METADATA_URI}`

Dieser Pfad gibt JSON-Metadaten für den Container zurück.

`${ECS_CONTAINER_METADATA_URI}/task`

Dieser Pfad gibt JSON-Metadaten für die Aufgabe zurück, einschließlich einer Liste der Container-IDs und Namen für alle Container, die der Aufgabe zugeordnet sind. Weitere Informationen zu der Antwort für diesen Endpunkt finden Sie unter Amazon ECS-Aufgabenmetadaten v3 JSON-Antwort für Aufgaben auf Fargate.

`${ECS_CONTAINER_METADATA_URI}/stats`

Dieser Pfad gibt JSON-Docker-Statistiken für die angegebene Docker-Container-ID zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Dieser Pfad gibt JSON-Docker-Statistiken für alle der Aufgabe zugeordneten Container zurück. Weitere Informationen zu den einzelnen zurückgegebenen Statistiken finden Sie ContainerStatsin der Docker-API-Dokumentation.

Amazon ECS-Aufgabenmetadaten v3 JSON-Antwort für Aufgaben auf Fargate

Die folgenden Informationen werden von der JSON-Antwort des Endpunkts für Aufgabenmetadaten (`${ECS_CONTAINER_METADATA_URI}/task`) zurückgegeben.

`Cluster`

Der Amazon-Ressourcenname (ARN) oder kurzer Name des Amazon-ECS-Clusters, zu dem die Aufgabe gehört.

`TaskARN`

Der vollständige Amazon-Ressourcenname (ARN) der Aufgabe, zu der der Container gehört

`Family`

Die Familie der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`Revision`

Die Revision der Amazon-ECS-Aufgabendefinition für die Aufgabe.

`DesiredStatus`

Der gewünschte Status der Aufgabe von Amazon ECS.

`KnownStatus`

Der bekannte Status der Aufgabe von Amazon ECS.

`Limits`

Die auf der Aufgabenebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`PullStartedAt`

Der Zeitstempel für den Zeitpunkt, an dem der erste Abruf des Container-Image gestartet wurde

`PullStoppedAt`

Der Zeitstempel für den Zeitpunkt, an dem der letzte Abruf des Container-Image abgeschlossen wurde

`AvailabilityZone`

Die Availability Zone, in der sich die Aufgabe befindet.

> **Note**
>
> Die Availability-Zone-Metadaten sind nur für Fargate-Aufgaben verfügbar, die Plattform Version 1.4 oder höher (Linux) oder 1.0.0 oder höher (Windows) verwenden.

`Containers`

Eine Liste der Container-Metadaten für jeden Container, der der Aufgabe zugeordnet ist

`DockerId`

Die Docker-ID des Containers

`Name`

Der Name des Containers, wie er in der Aufgabendefinition festgelegt ist

`DockerName`

Der Name des Containers, der vom Docker bereitgestellt wird. Der Amazon-ECS-Containeragent generiert einen eindeutigen Namen für den Container, um Namensüberschneidungen zu vermeiden, wenn mehrere Kopien derselben Aufgabendefinition auf einer einzelnen Instance ausgeführt werden.

`Image`

Das Image für den Container

`ImageID`

Der SHA-256-Digest für das Image

`Ports`

Alle für den Container zugänglichen Ports. Dieser Parameter wird ausgelassen, wenn keine zugänglichen Ports verfügbar sind.

`Labels`

Alle auf den Container angewendeten Kennungen. Dieser Parameter wird ausgelassen, wenn keine angewendeten Kennungen verfügbar sind.

`DesiredStatus`

Der gewünschte Status des Containers von Amazon ECS.

`KnownStatus`

Der bekannte Status des Containers von Amazon ECS.

`ExitCode`

Der Beendigungscode für den Container. Dieser Parameter wird ausgelassen, wenn der Container nicht beendet wurde.

`Limits`

Die auf der Containerebene festgelegten Ressourcenbegrenzungen, wie CPU (ausgedrückt in vCPUs) und Speicher. Dieser Parameter wird ausgelassen, wenn keine Ressourcenbegrenzungen festgelegt wurden.

`CreatedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container erstellt wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht erstellt wurde.

`StartedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container gestartet wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht gestartet wurde.

`FinishedAt`

Der Zeitstempel des Zeitpunkts, an dem der Container angehalten wurde. Dieser Parameter wird ausgelassen, wenn der Container noch nicht angehalten wurde.

`Type`

Der Typ des Containers. Container, die in der Aufgabendefinition festgelegt sind, haben den Typ `NORMAL`. Sie können andere Containertypen ignorieren, die für die interne Bereitstellung von Aufgabenressourcen durch den Amazon-ECS-Containeragenten verwendet werden.

`Networks`

Die Netzwerkinformationen für den Container, wie Netzwerkmodus und IP-Adresse. Dieser Parameter wird ausgelassen, wenn keine Netzwerkinformationen definiert sind.

`ClockDrift`

Die Informationen über die Abweichung zwischen der Referenzzeit und der Systemzeit. Dies gilt für das Linux-Betriebssystem. Diese Funktion verwendet den Amazon Time Sync Service, um die Genauigkeit der Uhr zu messen und den Zeitfehler für Container bereitzustellen. Weitere Informationen finden Sie unter Zeit für Ihre Linux-Instance festlegen im Amazon EC2 EC2-Benutzerhandbuch für Linux-Instances.

`ReferenceTime`

Die Grundlage der Taktgenauigkeit. Amazon ECS verwendet beispielsweise den globalen Standard Coordinated Universal Time (UTC) über NTP, z. B. `2021-09-07T16:57:44Z`.

`ClockErrorBound`

Das Maß des Taktfehlers, definiert als Abweichung zu UTC. Dieser Fehler ist der Unterschied in Millisekunden zwischen der Referenzzeit und der Systemzeit.

`ClockSynchronizationStatus`

Gibt an, ob der letzte Synchronisierungsversuch zwischen der Systemzeit und der Referenzzeit erfolgreich war.

Die gültigen Werte sind `SYNCHRONIZED` und `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Der Zeitstempel des Zeitpunkts, an dem der `DesiredStatus` der Aufgaben zu `STOPPED` gewechselt hat. Dies tritt auf, wenn ein entscheidender Container zu `STOPPED` wechselt.

Beispiele für Amazon ECS-Aufgabenmetadaten v3 für Aufgaben auf Fargate

Die folgende JSON-Antwort ist für eine Aufgabe mit einem Container.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
```

```
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Introspektion von Amazon ECS-Containern

Der Amazon-ECS-Container-Agent bietet eine API-Operation für das Erfassen von Details über die Container-Instance, auf der der Agent ausgeführt wird, sowie über die zugewiesenen Aufgaben, die auf dieser Instance ausgeführt werden. Sie können den Befehl curl innerhalb der Container-Instance verwenden, um den Amazon-ECS-Container-Agenten abzufragen (Port 51678) und zu den Container-Instance-Metadaten oder -Aufgabeninformationen zurückzukehren.

> **Important**
>
> Ihre Container-Instance muss über eine IAM-Rolle verfügen, die den Zugriff auf Amazon ECS erlaubt, um die Metadaten abzurufen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

Wenn Sie das Metadatenprotokoll der Container-Instance einsehen möchten, melden Sie sich über SSH bei Ihrer Container-Instance an und führen folgenden Befehl aus: Zu den Metadaten gehören die Container-Instance-ID, das Amazon-ECS-Cluster, in dem die Container-Instance registriert ist, und Informationen zur Amazon-ECS-Container-Agenten-Version.

```
curl -s http://localhost:51678/v1/metadata | python3 -mjson.tool
```

Ausgabe:

```
{
    "Cluster": "cluster_name",
    "ContainerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/cluster_name/container_instance_id",
    "Version": "Amazon ECS Agent - v1.30.0 (02ff320c)"
}
```

Wenn Sie Informationen über alle Aufgaben, die auf einer Container-Instance ausgeführt werden, einsehen möchten, melden Sie sich über SSH bei Ihrer Container-Instance an und führen folgenden Befehl aus:

```
curl http://localhost:51678/v1/tasks
```

Ausgabe:

```
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/example5-58ff-46c9-
ae05-543f8example",
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Family": "hello_world",
            "Version": "8",
            "Containers": [
                {
          "DockerId":
 "9581a69a761a557fbfce1d0f6745e4af5b9dbfb86b6b2c5c4df156f1a5932ff1",
          "DockerName": "ecs-hello_world-8-mysql-fcae8ac8f9f1d89d8301",
          "Name": "mysql",
          "CreatedAt": "2023-10-08T20:09:11.44527186Z",
          "StartedAt": "2023-10-08T20:09:11.44527186Z",
          "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        },
        {
          "DockerId":
 "bf25c5c5b2d4dba68846c7236e75b6915e1e778d31611e3c6a06831e39814a15",
          "DockerName": "ecs-hello_world-8-wordpress-e8bfddf9b488dff36c00",
          "Name": "wordpress"
        }
    ]
}
    ]
}
```

Sie können Informationen zu einer bestimmten Aufgabe, die auf einer Container-Instance ausgeführt wird, abrufen. Um eine spezifische Aufgabe oder einen Container zu bestimmen, fügen Sie der Anfrage Folgendes hinzu:

- Aufgaben-ARN (`?taskarn=`*`task_arn`*)
- Die Docker-ID für einen Container (`?dockerid=`*`docker_id`*)

Um Aufgabeninformationen mit einer Docker-ID für Container abzurufen, melden Sie sich über SSH bei Ihrer Container-Instance an und führen folgenden Befehl aus.

> **Note**
>
> Amazon-ECS-Container-Agenten vor Version 1.14.2 erfordern für die Introspektions-API komplette Docker-Container-IDs, nicht die verkürzte Version, die bei docker ps angezeigt wird. Sie können die komplette Docker-ID für einen Container abrufen, indem Sie den Befehl docker ps --no-trunc auf der Container-Instance ausführen.

```
curl http://localhost:51678/v1/tasks?dockerid=79c796ed2a7f
```

Ausgabe:

```
{
    "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/e01d58a8-151b-40e8-
bc01-22647b9ecfec",
    "Containers": [
        {
            "DockerId":
 "79c796ed2a7f864f485c76f83f3165488097279d296a7c05bd5201a1c69b2920",
            "DockerName": "ecs-nginx-efs-2-nginx-9ac0808dd0afa495f001",
            "Name": "nginx",
            "CreatedAt": "2023-10-08T20:09:11.44527186Z",
            "StartedAt": "2023-10-08T20:09:11.44527186Z",
            "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        }
    ],
    "DesiredStatus": "RUNNING",
    "Family": "nginx-efs",
    "KnownStatus": "RUNNING",
    "Version": "2"
}
```

Identifizieren Sie mithilfe von Runtime Monitoring nicht autorisiertes Verhalten

Amazon GuardDuty ist ein Service zur Bedrohungserkennung, der Ihnen hilft, Ihre Konten, Container, Workloads und die Daten in Ihrer AWS Umgebung zu schützen. Mithilfe von Modellen für maschinelles Lernen (ML) und Funktionen zur Erkennung von Anomalien und Bedrohungen werden GuardDuty kontinuierlich verschiedene Protokollquellen und Laufzeitaktivitäten überwacht, um potenzielle Sicherheitsrisiken und böswillige Aktivitäten in Ihrer Umgebung zu identifizieren und zu priorisieren.

Runtime Monitoring in GuardDuty schützt Workloads, die auf Fargate- und EC2-Container-Instances ausgeführt werden, indem die AWS Protokoll- und Netzwerkaktivitäten kontinuierlich überwacht werden, um bösartiges oder nicht autorisiertes Verhalten zu erkennen. Runtime Monitoring verwendet einen schlanken, vollständig verwalteten GuardDuty Security Agent, der das Verhalten auf dem Host analysiert, z. B. den Dateizugriff, die Prozessausführung und Netzwerkverbindungen. Dies deckt Probleme wie die Eskalation von Rechten, die Verwendung offengelegter Anmeldeinformationen oder die Kommunikation mit bösartigen IP-Adressen, Domänen und das Vorhandensein von Malware auf Ihren Amazon EC2 EC2-Instances und Container-Workloads ab. Weitere Informationen finden Sie unter GuardDutyRuntime Monitoring im GuardDuty Benutzerhandbuch.

Ihr Sicherheitsadministrator aktiviert Runtime Monitoring für ein einzelnes oder mehrere Konten in AWS Organizations für GuardDuty. Sie wählen auch aus, ob der GuardDuty Security Agent GuardDuty automatisch installiert wird, wenn Sie Fargate verwenden. Alle Ihre Cluster sind automatisch geschützt und GuardDuty verwalten den Security Agent in Ihrem Namen.

In den folgenden Fällen können Sie den GuardDuty Security Agent auch manuell konfigurieren:

- Sie verwenden EC2-Container-Instances
- Sie benötigen eine detaillierte Steuerung, um Runtime Monitoring auf Cluster-Ebene zu aktivieren

Um Runtime Monitoring verwenden zu können, müssen Sie die geschützten Cluster konfigurieren und den GuardDuty Security Agent auf Ihren EC2-Container-Instances installieren und verwalten.

So funktioniert Runtime Monitoring mit Amazon ECS

Runtime Monitoring verwendet einen schlanken GuardDuty Sicherheitsagenten, der die Amazon ECS-Workload-Aktivitäten im Hinblick darauf überwacht, wie Anwendungen die zugrunde liegenden Systemressourcen anfordern, darauf zugreifen und diese verbrauchen.

Bei Fargate-Aufgaben wird der GuardDuty Security Agent als Sidecar-Container für jede Aufgabe ausgeführt.

Bei EC2-Container-Instances wird der GuardDuty Security Agent als Prozess auf der Instance ausgeführt.

Der GuardDuty Security Agent sammelt Daten aus den folgenden Ressourcen und sendet die Daten dann zur Verarbeitung GuardDuty an. Sie können die Ergebnisse in der GuardDuty Konsole einsehen. Sie können sie auch an andere Sicherheitsanbieter AWS Security Hub, AWS-Services z. B. oder einen Drittanbieter für Sicherheitslösungen, zur Aggregation und Problembehebung senden. Informationen zum Anzeigen und Verwalten von Ergebnissen finden Sie unter GuardDuty Amazon-Ergebnisse verwalten im GuardDuty Amazon-Benutzerhandbuch.

- Antworten aus den folgenden Amazon ECS-API-Aufrufen:
  - DescribeClusters

    Zu den Antwortparametern gehört das Runtime Monitoring-Tag (wenn das Tag gesetzt ist), wenn Sie die `--include TAGS` Option verwenden.
  - DescribeTasks

    Für den Starttyp Fargate beinhalten die Antwortparameter den GuardDuty Sidecar-Container.
  - ListAccountSettings

    Zu den Antwortparametern gehört die Runtime Monitoring-Kontoeinstellung, die von Ihrem Sicherheitsadministrator festgelegt wird.
- Die Introspektionsdaten des Container-Agenten. Weitere Informationen finden Sie unter Introspektion von Amazon ECS-Containern.
- Der Endpunkt der Aufgabenmetadaten für den Starttyp:
  - Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4
  - Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4, für Aufgaben auf Fargate

Überlegungen

Beachten Sie bei der Verwendung von Runtime Monitoring Folgendes:

- Runtime Monitoring ist mit Kosten verbunden. Weitere Informationen finden Sie unter GuardDuty Amazon-Preise.

- Runtime Monitoring wird auf Amazon ECS Anywhere nicht unterstützt.
- Runtime Monitoring wird für das Windows-Betriebssystem nicht unterstützt.
- Wenn Sie Amazon ECS Exec auf Fargate verwenden, müssen Sie den Container-Namen angeben, da der GuardDuty Security Agent als Sidecar-Container läuft.
- Sie können Amazon ECS Exec nicht auf dem Sidecar-Container des GuardDuty Security Agents verwenden.
- Der IAM-Benutzer, der Runtime Monitoring auf Cluster-Ebene steuert, muss über die entsprechenden IAM-Berechtigungen für das Tagging verfügen. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent, der in einem privaten Amazon ECR-Repository gespeichert ist, in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten.

Ressourcenauslastung

Das Tag, das Sie dem Cluster hinzufügen, wird auf das Cluster-Tag-Kontingent angerechnet.

Der GuardDuty Agent-Sidecar-Container wird nicht auf das Kontingent für Container pro Aufgabendefinition angerechnet.

Wie bei der meisten Sicherheitssoftware gibt es einen leichten Mehraufwand für. GuardDuty Informationen zu den Fargate-Speicherlimits finden Sie unter CPU- und Speicherlimits im GuardDuty Benutzerhandbuch. Informationen zu den Amazon EC2 EC2-Speicherlimits finden Sie unter CPU- und Speicherlimit für GuardDuty Agenten.

Laufzeitüberwachung für Amazon ECS Fargate-Workloads

Wenn Sie EC2-Container-Instances verwenden, müssen Sie Runtime Monitoring manuell konfigurieren. Weitere Informationen finden Sie unter Laufzeitüberwachung für EC2-Workloads auf Amazon ECS.

Sie können den Security Agent auf Ihren Container-Instances GuardDuty verwalten lassen. Diese Option ist nur für Fargate verfügbar. Diese Option (GuardDuty Agentenverwaltung) ist verfügbar in GuardDuty

Wenn Sie die GuardDuty Agentenverwaltung verwenden, GuardDuty werden die folgenden Operationen ausgeführt:

- Erstellt VPC-Endpunkte GuardDuty für jede VPC, die einen Cluster hostet.
- Ruft den neuesten GuardDuty Security Agent ab und installiert ihn als Sidecar-Container für alle neuen eigenständigen Fargate-Aufgaben und neuen Servicebereitstellungen.

  Eine neue Dienstbereitstellung erfolgt, wenn Sie einen Dienst zum ersten Mal starten oder wenn Sie einen vorhandenen Dienst mit der Option Neue Bereitstellung erzwingen aktualisieren.

Runtime Monitoring für Amazon ECS einschalten

Sie können so konfigurieren GuardDuty , dass der Security Agent für alle Ihre Fargate-Cluster automatisch verwaltet wird.

Die folgenden Voraussetzungen müssen für die Verwendung von Runtime Monitoring erfüllt sein:

- Die Fargate-Plattformversion muss für Linux `1.4.0` oder höher sein.
- IAM-Rollen und -Berechtigungen für Amazon ECS:
  - Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.
  - Sie steuern die Laufzeitüberwachung für einen Cluster mit einem vordefinierten Tag. Wenn Ihre Zugriffsrichtlinien den Zugriff auf der Grundlage von Tags einschränken, müssen Sie Ihren IAM-Benutzern explizite Berechtigungen zum Taggen von Clustern gewähren. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Verbindung zum Amazon ECR-Repository herstellen:

  Der GuardDuty Security Agent ist in einem Amazon ECR-Repository gespeichert. Jede Standalone- und Service-Task muss Zugriff auf das Repository haben. Verwenden Sie eine der folgenden Optionen:

  - Für Aufgaben in öffentlichen Subnetzen können Sie entweder eine öffentliche IP-Adresse für die Aufgabe verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.

  - Für Aufgaben in privaten Subnetzen können Sie ein Network Address Translation (NAT) - Gateway verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird.

    Weitere Informationen finden Sie unter Verwenden eines privaten Subnetzes und eines NAT-Gateways.
- Sie müssen die `AWSServiceRoleForAmazonGuardDuty` Rolle für GuardDuty haben. Weitere Informationen finden Sie unter Servicebezogene Rollenberechtigungen für GuardDuty im GuardDutyAmazon-Benutzerhandbuch.
- Alle Dateien, die Sie mit Runtime Monitoring schützen möchten, müssen für den Root-Benutzer zugänglich sein. Wenn Sie die Berechtigungen einer Datei manuell geändert haben, müssen Sie sie auf setzen`755`.

Die folgenden Voraussetzungen gelten für die Verwendung von Runtime Monitoring auf EC2-Container-Instances:

- Sie müssen Version `20230929` oder höher des Amazon ECS-AMI verwenden.
- Sie müssen Amazon ECS Agent auf Version `1.77` oder höher auf den Container-Instances ausführen.
- Sie müssen die Kernel-Version `5.10` oder höher verwenden.
- Informationen zu den unterstützten Linux-Betriebssystemen und Architekturen finden Sie unter Welche Betriebsmodelle und Workloads unterstützt GuardDuty Runtime Monitoring.
- Sie können Systems Manager verwenden, um Ihre Container-Instances zu verwalten. Weitere Informationen finden Sie im AWS Systems Manager Session Manager Benutzerhandbuch unter Systems Manager für EC2-Instances einrichten.

Sie aktivieren Runtime Monitoring in GuardDuty. Informationen zur Aktivierung der Funktion finden Sie unter Enabling Runtime Monitoring im GuardDuty Amazon-Benutzerhandbuch.

Hinzufügen von Runtime Monitoring zu bestehenden Amazon ECS Fargate-Aufgaben

Wenn Sie Runtime Monitoring aktivieren, werden alle neuen eigenständigen Aufgaben und neuen Dienstbereitstellungen im Cluster automatisch geschützt. Um die Unveränderbarkeitsbeschränkung beizubehalten, sind bestehende Aufgaben nicht betroffen.

Die folgenden Voraussetzungen müssen für die Verwendung von Runtime Monitoring erfüllt sein:

- Die Fargate-Plattformversion muss für Linux `1.4.0` oder höher sein.
- IAM-Rollen und -Berechtigungen für Amazon ECS:
  - Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.
  - Sie steuern die Laufzeitüberwachung für einen Cluster mit einem vordefinierten Tag. Wenn Ihre Zugriffsrichtlinien den Zugriff auf der Grundlage von Tags einschränken, müssen Sie Ihren IAM-Benutzern explizite Berechtigungen zum Taggen von Clustern gewähren. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Verbindung zum Amazon ECR-Repository herstellen:

  Der GuardDuty Security Agent ist in einem Amazon ECR-Repository gespeichert. Jede Standalone- und Service-Task muss Zugriff auf das Repository haben. Verwenden Sie eine der folgenden Optionen:
  - Für Aufgaben in öffentlichen Subnetzen können Sie entweder eine öffentliche IP-Adresse für die Aufgabe verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Für Aufgaben in privaten Subnetzen können Sie ein Network Address Translation (NAT) - Gateway verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird.

    Weitere Informationen finden Sie unter Verwenden eines privaten Subnetzes und eines NAT-Gateways.
- Sie müssen die `AWSServiceRoleForAmazonGuardDuty` Rolle für GuardDuty haben. Weitere Informationen finden Sie unter Servicebezogene Rollenberechtigungen für GuardDuty im GuardDutyAmazon-Benutzerhandbuch.
- Alle Dateien, die Sie mit Runtime Monitoring schützen möchten, müssen für den Root-Benutzer zugänglich sein. Wenn Sie die Berechtigungen einer Datei manuell geändert haben, müssen Sie sie auf setzen`755`.

Die folgenden Voraussetzungen gelten für die Verwendung von Runtime Monitoring auf EC2-Container-Instances:

- Sie müssen Version `20230929` oder höher des Amazon ECS-AMI verwenden.
- Sie müssen Amazon ECS Agent auf Version `1.77` oder höher auf den Container-Instances ausführen.
- Sie müssen die Kernel-Version `5.10` oder höher verwenden.
- Informationen zu den unterstützten Linux-Betriebssystemen und Architekturen finden Sie unter Welche Betriebsmodelle und Workloads unterstützt GuardDuty Runtime Monitoring.
- Sie können Systems Manager verwenden, um Ihre Container-Instances zu verwalten. Weitere Informationen finden Sie im AWS Systems Manager Session Manager Benutzerhandbuch unter Systems Manager für EC2-Instances einrichten.

Um eine Aufgabe sofort zu schützen, müssen Sie eine der folgenden Aktionen ausführen:

- Stoppen Sie bei eigenständigen Aufgaben die Aufgaben und starten Sie sie dann. Weitere Informationen finden Sie unter Eine Amazon ECS-Aufgabe beenden und Eine Anwendung als Amazon ECS-Aufgabe ausführen
- Bei Aufgaben, die Teil eines Dienstes sind, aktualisieren Sie den Dienst mit der Option „Neue Bereitstellung erzwingen“. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.

Runtime Monitoring aus einem Amazon ECS-Cluster entfernen

Möglicherweise möchten Sie bestimmte Cluster vom Schutz ausschließen, z. B. Cluster, die Sie für Tests verwenden. Dies führt GuardDuty dazu, dass die folgenden Operationen mit Ressourcen im Cluster ausgeführt werden:

- Setzen Sie den GuardDuty Security Agent nicht mehr für neue eigenständige Fargate-Aufgaben oder neue Serviceeinsätze ein.

  Um die Einschränkung der Unveränderlichkeit aufrechtzuerhalten, sind bestehende Aufgaben und Bereitstellungen, bei denen Runtime Monitoring aktiviert ist, nicht betroffen.
- Beendet die Fakturierung und akzeptiert keine Laufzeitereignisse mehr für Aufgaben.

Gehen Sie wie folgt vor, um Runtime Monitoring aus einem Cluster zu entfernen.

1. Verwenden Sie die Amazon ECS-Konsole oder AWS CLI setzen Sie den `GuardDutyManaged` Tag-Schlüssel auf dem Cluster auf`false`. Weitere Informationen finden Sie unter Aktualisieren eines Clusters oder Arbeiten mit Tags mithilfe der CLI oder API. Verwenden Sie die folgenden Werte für das Tag.

   > **Note**
   >
   > Bei Schlüssel und Wert wird zwischen Groß- und Kleinschreibung unterschieden und sie müssen genau mit den Zeichenketten übereinstimmen.

   Schlüssel =`GuardDutyManaged`, Wert = `false`
2. Löschen Sie den GuardDuty VPC-Endpunkt für den Cluster. Weitere Informationen zum Löschen von VPC-Endpunkten finden Sie unter Löschen eines Schnittstellenendpunkts im AWS PrivateLink Benutzerhandbuch.

Runtime Monitoring for Amazon ECS von einem Konto entfernen

Wenn Sie Runtime Monitoring nicht mehr verwenden möchten, deaktivieren Sie die Funktion in GuardDuty. Informationen zur Deaktivierung der Funktion finden Sie unter Enabling Runtime Monitoring im GuardDuty Amazon-Benutzerhandbuch.

GuardDuty führt die folgenden Operationen aus:

- Löscht die VPC-Endpunkte GuardDuty für jede VPC, die einen Cluster hostet.
- Der GuardDuty Security Agent wird nicht mehr für neue eigenständige Fargate-Aufgaben oder neue Serviceeinsätze eingesetzt.

  Um die Einschränkung der Unveränderlichkeit aufrechtzuerhalten, werden bestehende Aufgaben und Bereitstellungen erst beeinträchtigt, wenn sie gestoppt, repliziert oder skaliert werden.
- Stoppt die Abrechnung und akzeptiert keine Laufzeitereignisse mehr für Aufgaben.

Laufzeitüberwachung für EC2-Workloads auf Amazon ECS

Verwenden Sie diese Option, wenn Sie EC2-Instances für Ihre Kapazität verwenden oder wenn Sie eine detaillierte Steuerung der Runtime Monitoring auf Cluster-Ebene auf Fargate benötigen.

Sie stellen die Cluster für Runtime Monitoring bereit, indem Sie ein vordefiniertes Tag hinzufügen.

Für EC2-Container-Instances laden Sie den GuardDuty Security Agent herunter, installieren und verwalten ihn.

Für Fargate GuardDuty leitet er den Sicherheitsagenten in Ihrem Namen.

Runtime Monitoring für Amazon ECS einschalten

Sie können Runtime Monitoring für Cluster mit EC2-Instances aktivieren oder wenn Sie eine detaillierte Steuerung der Runtime Monitoring auf Cluster-Ebene auf Fargate benötigen.

Die folgenden Voraussetzungen müssen für die Verwendung von Runtime Monitoring erfüllt sein:

- Die Fargate-Plattformversion muss für Linux `1.4.0` oder höher sein.
- IAM-Rollen und -Berechtigungen für Amazon ECS:
  - Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.
  - Sie steuern die Laufzeitüberwachung für einen Cluster mit einem vordefinierten Tag. Wenn Ihre Zugriffsrichtlinien den Zugriff auf der Grundlage von Tags einschränken, müssen Sie Ihren IAM-Benutzern explizite Berechtigungen zum Taggen von Clustern gewähren. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Verbindung zum Amazon ECR-Repository herstellen:

  Der GuardDuty Security Agent ist in einem Amazon ECR-Repository gespeichert. Jede Standalone- und Service-Task muss Zugriff auf das Repository haben. Verwenden Sie eine der folgenden Optionen:

  - Für Aufgaben in öffentlichen Subnetzen können Sie entweder eine öffentliche IP-Adresse für die Aufgabe verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Für Aufgaben in privaten Subnetzen können Sie ein Network Address Translation (NAT) - Gateway verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird.

Weitere Informationen finden Sie unter Verwenden eines privaten Subnetzes und eines NAT-Gateways.

- Sie müssen die `AWSServiceRoleForAmazonGuardDuty` Rolle für GuardDuty haben. Weitere Informationen finden Sie unter Servicebezogene Rollenberechtigungen für GuardDuty im GuardDutyAmazon-Benutzerhandbuch.
- Alle Dateien, die Sie mit Runtime Monitoring schützen möchten, müssen für den Root-Benutzer zugänglich sein. Wenn Sie die Berechtigungen einer Datei manuell geändert haben, müssen Sie sie auf setzen`755`.

Die folgenden Voraussetzungen gelten für die Verwendung von Runtime Monitoring auf EC2-Container-Instances:

- Sie müssen Version `20230929` oder höher des Amazon ECS-AMI verwenden.
- Sie müssen Amazon ECS Agent auf Version `1.77` oder höher auf den Container-Instances ausführen.
- Sie müssen die Kernel-Version `5.10` oder höher verwenden.
- Informationen zu den unterstützten Linux-Betriebssystemen und Architekturen finden Sie unter Welche Betriebsmodelle und Workloads unterstützt GuardDuty Runtime Monitoring.
- Sie können Systems Manager verwenden, um Ihre Container-Instances zu verwalten. Weitere Informationen finden Sie im AWS Systems Manager Session Manager Benutzerhandbuch unter Systems Manager für EC2-Instances einrichten.

Sie aktivieren Runtime Monitoring in GuardDuty. Informationen zur Aktivierung der Funktion finden Sie unter Enabling Runtime Monitoring im GuardDuty Amazon-Benutzerhandbuch.

Hinzufügen von Runtime Monitoring zu einem Amazon ECS-Cluster

Konfigurieren Sie Runtime Monitoring für den Cluster und installieren Sie dann den GuardDuty Security Agent auf Ihren EC2-Container-Instances.

Die folgenden Voraussetzungen müssen für die Verwendung von Runtime Monitoring erfüllt sein:

- Die Fargate-Plattformversion muss für Linux `1.4.0` oder höher sein.
- IAM-Rollen und -Berechtigungen für Amazon ECS:

  - Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.
  - Sie steuern die Laufzeitüberwachung für einen Cluster mit einem vordefinierten Tag. Wenn Ihre Zugriffsrichtlinien den Zugriff auf der Grundlage von Tags einschränken, müssen Sie Ihren IAM-Benutzern explizite Berechtigungen zum Taggen von Clustern gewähren. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Verbindung zum Amazon ECR-Repository herstellen:

  Der GuardDuty Security Agent ist in einem Amazon ECR-Repository gespeichert. Jede Standalone- und Service-Task muss Zugriff auf das Repository haben. Verwenden Sie eine der folgenden Optionen:

  - Für Aufgaben in öffentlichen Subnetzen können Sie entweder eine öffentliche IP-Adresse für die Aufgabe verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Für Aufgaben in privaten Subnetzen können Sie ein Network Address Translation (NAT) - Gateway verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird.

    Weitere Informationen finden Sie unter Verwenden eines privaten Subnetzes und eines NAT-Gateways.
- Sie müssen die `AWSServiceRoleForAmazonGuardDuty` Rolle für GuardDuty haben. Weitere Informationen finden Sie unter Servicebezogene Rollenberechtigungen für GuardDuty im GuardDutyAmazon-Benutzerhandbuch.
- Alle Dateien, die Sie mit Runtime Monitoring schützen möchten, müssen für den Root-Benutzer zugänglich sein. Wenn Sie die Berechtigungen einer Datei manuell geändert haben, müssen Sie sie auf setzen`755`.

Die folgenden Voraussetzungen gelten für die Verwendung von Runtime Monitoring auf EC2-Container-Instances:

- Sie müssen Version `20230929` oder höher des Amazon ECS-AMI verwenden.

- Sie müssen Amazon ECS Agent auf Version `1.77` oder höher auf den Container-Instances ausführen.
- Sie müssen die Kernel-Version `5.10` oder höher verwenden.
- Informationen zu den unterstützten Linux-Betriebssystemen und Architekturen finden Sie unter Welche Betriebsmodelle und Workloads unterstützt GuardDuty Runtime Monitoring.
- Sie können Systems Manager verwenden, um Ihre Container-Instances zu verwalten. Weitere Informationen finden Sie im AWS Systems Manager Session Manager Benutzerhandbuch unter Systems Manager für EC2-Instances einrichten.

Führen Sie die folgenden Vorgänge aus, um Runtime Monitoring zu einem Cluster hinzuzufügen.

1. Erstellen Sie einen VPC-Endpunkt GuardDuty für jede Cluster-VPC. Weitere Informationen finden Sie unter Manuelles Erstellen eines Amazon VPC-Endpunkts im GuardDuty Benutzerhandbuch.
2. Konfigurieren Sie die EC2-Container-Instances.

   a. Aktualisieren Sie den Amazon ECS-Agenten auf Version `1.77` oder höher auf den EC2-Container-Instances im Cluster. Weitere Informationen finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

   b. Installieren Sie den GuardDuty Security Agent auf den EC2-Container-Instances im Cluster. Weitere Informationen finden Sie unter Manuelles Verwalten des Security Agents auf einer Amazon EC2 EC2-Instance im GuardDuty Benutzerhandbuch.

   Alle neuen und vorhandenen Aufgaben und Bereitstellungen sind sofort geschützt, da der GuardDuty Security Agent als Prozess auf der EC2-Container-Instance ausgeführt wird.
3. Verwenden Sie die Amazon ECS-Konsole oder AWS CLI setzen Sie den `GuardDutyManaged` Tag-Schlüssel auf dem Cluster auf`true`. Weitere Informationen finden Sie unter Aktualisieren eines Clusters oder Arbeiten mit Tags mithilfe der CLI oder API. Verwenden Sie die folgenden Werte für das Tag.

> **Note**
>
> Bei Schlüssel und Wert wird zwischen Groß- und Kleinschreibung unterschieden und sie müssen genau mit den Zeichenketten übereinstimmen.

Schlüssel =`GuardDutyManaged`, Wert = `true`

Hinzufügen von Runtime Monitoring zu bestehenden Amazon ECS-Aufgaben

Wenn Sie Runtime Monitoring aktivieren, werden alle neuen eigenständigen Aufgaben und neuen Dienstbereitstellungen im Cluster automatisch geschützt. Um die Unveränderbarkeitsbeschränkung beizubehalten, sind bestehende Aufgaben nicht betroffen.

Die folgenden Voraussetzungen müssen für die Verwendung von Runtime Monitoring erfüllt sein:

- Die Fargate-Plattformversion muss für Linux `1.4.0` oder höher sein.
- IAM-Rollen und -Berechtigungen für Amazon ECS:
  - Fargate-Aufgaben müssen eine Aufgabenausführungsrolle verwenden. Diese Rolle erteilt den Aufgaben die Berechtigung, den GuardDuty Security Agent in Ihrem Namen abzurufen, zu aktualisieren und zu verwalten. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.
  - Sie steuern die Laufzeitüberwachung für einen Cluster mit einem vordefinierten Tag. Wenn Ihre Zugriffsrichtlinien den Zugriff auf der Grundlage von Tags einschränken, müssen Sie Ihren IAM-Benutzern explizite Berechtigungen zum Taggen von Clustern gewähren. Weitere Informationen finden Sie im IAM-Tutorial: Definieren von Berechtigungen für den Zugriff auf AWS Ressourcen auf der Grundlage von Tags im IAM-Benutzerhandbuch.
- Verbindung zum Amazon ECR-Repository herstellen:

  Der GuardDuty Security Agent ist in einem Amazon ECR-Repository gespeichert. Jede Standalone- und Service-Task muss Zugriff auf das Repository haben. Verwenden Sie eine der folgenden Optionen:

  - Für Aufgaben in öffentlichen Subnetzen können Sie entweder eine öffentliche IP-Adresse für die Aufgabe verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird. Weitere Informationen finden Sie unter Amazon ECR-Schnittstellen VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.
  - Für Aufgaben in privaten Subnetzen können Sie ein Network Address Translation (NAT) - Gateway verwenden oder einen VPC-Endpunkt für Amazon ECR in dem Subnetz erstellen, in dem die Aufgabe ausgeführt wird.

Weitere Informationen finden Sie unter Verwenden eines privaten Subnetzes und eines NAT-Gateways.

- Sie müssen die `AWSServiceRoleForAmazonGuardDuty` Rolle für GuardDuty haben. Weitere Informationen finden Sie unter Servicebezogene Rollenberechtigungen für GuardDuty im GuardDutyAmazon-Benutzerhandbuch.
- Alle Dateien, die Sie mit Runtime Monitoring schützen möchten, müssen für den Root-Benutzer zugänglich sein. Wenn Sie die Berechtigungen einer Datei manuell geändert haben, müssen Sie sie auf setzen`755`.

Die folgenden Voraussetzungen gelten für die Verwendung von Runtime Monitoring auf EC2-Container-Instances:

- Sie müssen Version `20230929` oder höher des Amazon ECS-AMI verwenden.
- Sie müssen Amazon ECS Agent auf Version `1.77` oder höher auf den Container-Instances ausführen.
- Sie müssen die Kernel-Version `5.10` oder höher verwenden.
- Informationen zu den unterstützten Linux-Betriebssystemen und Architekturen finden Sie unter Welche Betriebsmodelle und Workloads unterstützt GuardDuty Runtime Monitoring.
- Sie können Systems Manager verwenden, um Ihre Container-Instances zu verwalten. Weitere Informationen finden Sie im AWS Systems Manager Session Manager Benutzerhandbuch unter Systems Manager für EC2-Instances einrichten.

Um eine Aufgabe sofort zu schützen, müssen Sie eine der folgenden Aktionen ausführen:

- Stoppen Sie bei eigenständigen Aufgaben die Aufgaben und starten Sie sie dann. Weitere Informationen finden Sie unter Eine Amazon ECS-Aufgabe beenden und Eine Anwendung als Amazon ECS-Aufgabe ausführen
- Bei Aufgaben, die Teil eines Dienstes sind, aktualisieren Sie den Dienst mit der Option „Neue Bereitstellung erzwingen“. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole.

Runtime Monitoring aus einem Amazon ECS-Cluster entfernen

Sie können Runtime Monitoring aus einem Cluster entfernen. Dies führt GuardDuty dazu, dass die Überwachung aller Ressourcen im Cluster beendet wird.

Um Runtime Monitoring aus einem Cluster zu entfernen.

1. Verwenden Sie die Amazon ECS-Konsole oder AWS CLI setzen Sie den `GuardDutyManaged` Tag-Schlüssel auf dem Cluster auf`false`. Weitere Informationen finden Sie unter Aktualisieren eines Clusters oder Arbeiten mit Tags mithilfe der CLI oder API.

   > **Note**
   >
   > Bei Schlüssel und Wert wird zwischen Groß- und Kleinschreibung unterschieden und sie müssen exakt mit den Zeichenketten übereinstimmen.

   Schlüssel =`GuardDutyManaged`, Wert = `false`
2. Deinstallieren Sie den GuardDuty Security Agent auf Ihren EC2-Container-Instances im Cluster.

   Weitere Informationen finden Sie unter Manuelles Deinstallieren des Security Agents im GuardDuty Benutzerhandbuch.
3. Löschen Sie den GuardDuty VPC-Endpunkt für jede Cluster-VPC. Weitere Informationen zum Löschen von VPC-Endpunkten finden Sie unter Löschen eines Schnittstellenendpunkts im AWS PrivateLink Benutzerhandbuch.

Den GuardDuty Security Agent auf Ihren Amazon ECS-Container-Instances aktualisieren

Informationen zum Aktualisieren des GuardDuty Security Agents auf Ihren EC2-Container-Instances finden Sie unter Aktualisieren des GuardDuty Security Agents im GuardDuty Amazon-Benutzerhandbuch.

Runtime Monitoring for Amazon ECS von einem Konto entfernen

Wenn Sie Runtime Monitoring nicht mehr verwenden möchten, deaktivieren Sie die Funktion in GuardDuty. Informationen zur Deaktivierung der Funktion finden Sie unter Enabling Runtime Monitoring im GuardDuty Amazon-Benutzerhandbuch.

Entfernen Sie Runtime Monitoring aus allen Clustern. Weitere Informationen finden Sie unter Runtime Monitoring aus einem Amazon ECS-Cluster entfernen.

Häufig gestellte Fragen zur Fehlerbehebung bei Runtime Monitoring

Möglicherweise müssen Sie Fehler beheben oder überprüfen, ob Runtime Monitoring aktiviert ist und für Ihre Aufgaben und Container ausgeführt wird.

Themen

Woran erkenne ich, ob Runtime Monitoring in meinem Konto aktiv ist?

In der Amazon ECS-Konsole befinden sich die Informationen auf der Seite Kontoeinstellungen.

Sie können die `effective-settings` Option `list-account-settings` auch verwenden.

```
aws ecs list-account-settings --effective-settings
```

Output

Die Einstellung, bei der der Name auf `guardDutyActivate` und der Wert auf gesetzt ist, on gibt an, dass das Konto konfiguriert ist. Sie müssen sich bei Ihrem GuardDuty Administrator erkundigen, ob die Verwaltung automatisch oder manuell erfolgt.

```
{
    "setting": {
```

```
        "name": "guardDutyActivate",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": "aws-managed"
    }
}
```

Woran erkenne ich, ob Runtime Monitoring auf einem Cluster aktiv ist?

In der Amazon ECS-Konsole befinden sich die Informationen auf der Cluster-Detailseite auf der Registerkarte „Tags“.

Sie können die `TAGS` Option `describe-clusters` auch verwenden.

Das folgende Beispiel zeigt die Ausgabe für den Standardcluster

```
aws ecs describe-clusters --cluster default --include TAGS
```

Output

Das Tag, bei dem Key auf `GuardDutyManaged` und Value auf gesetzt ist, `true` gibt an, dass der Cluster für Runtime Monitoring konfiguriert ist.

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-east-1:1234567890:cluster/default",
            "clusterName": "default",
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 1,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [
                {
                    "key": "GuardDutyManaged",
                    "value": "true"
                }
            ],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
```

```
        }
    ],
    "failures": []
}
```

Woran erkenne ich, ob der GuardDuty Security Agent auf einer Fargate-Aufgabe läuft?

Der GuardDuty Security Agent läuft als Sidecar-Container für Fargate-Aufgaben.

In der Amazon ECS-Konsole wird der Sidecar unter Container auf der Seite mit den Aufgabendetails angezeigt.

Sie können den Container ausführen `describe-tasks` und nach ihm suchen, dessen Name auf `aws-gd-agent` und lastStatus auf gesetzt sind. `RUNNING`

Das folgende Beispiel zeigt die Ausgabe für den Standardcluster für Aufgaben`aws:ecs:us-east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE`.

```
aws ecs describe-tasks --cluster default --tasks aws:ecs:us-
east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE
```

Output

Der angegebene Container `gd-agent` befindet sich im `RUNNING` Status.

```
"containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-
f057-467b-a079-96167EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/0b69d5c0-
d655-4695-98cd-5d2d5EXAMPLE",
        "lastStatus": "RUNNING",
        "healthStatus": "UNKNOWN",
        "memory": "string",
        "name": "aws-gd-agent"
      }
    ]
```

Woran erkenne ich, ob der GuardDuty Security Agent auf einer EC2-Container-Instance läuft?

Führen Sie den folgenden Befehl aus, um den Status anzuzeigen:

```
sudo systemctl status amazon-guardduty-agent
```

Die Protokolldatei befindet sich am folgenden Speicherort:

```
/var/log/amzn-guardduty-agent
```

Was passiert, wenn es für eine Aufgabe, die auf dem Cluster ausgeführt wird, keine Rolle zur Aufgabenausführung gibt?

Bei Fargate-Aufgaben wird die Aufgabe ohne den Sidecar-Container des GuardDuty Security Agents gestartet. Im GuardDuty Dashboard mit den Deckungsstatistiken wird angezeigt, dass für die Aufgabe kein Schutz vorhanden ist.

Woran erkenne ich, ob ich über die richtigen Berechtigungen zum Taggen von Clustern für Runtime Monitoring verfüge?

Um einen Cluster zu taggen, müssen Sie die `ecs:TagResource` Aktion `CreateCluster` sowohl für als auch haben`UpdateCluster`.

Im Folgenden finden Sie einen Auszug aus einer Beispielrichtlinie.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
             "ecs:CreateAction" : "CreateCluster",
             "ecs:CreateAction" : "UpdateCluster",
          }
       }
    }
  ]
}
```

Was passiert, wenn keine Verbindung zu Amazon ECR besteht?

Bei Fargate-Aufgaben wird die Aufgabe ohne den Sidecar-Container des GuardDuty Security Agents gestartet. Im GuardDuty Dashboard mit den Deckungsstatistiken wird angezeigt, dass für die Aufgabe kein Schutz vorhanden ist.

Wie behebe ich Speicherfehler bei meinen Fargate-Aufgaben, nachdem ich Runtime Monitoring aktiviert habe?

Der GuardDuty Security Agent ist ein einfacher Prozess. Der Prozess verbraucht jedoch je nach Größe der Arbeitslast immer noch Ressourcen. Wir empfehlen, Tools zur Nachverfolgung von Container-Ressourcen wie Amazon CloudWatch Container Insights zu verwenden, um GuardDuty Bereitstellungen in Ihrem Cluster bereitzustellen. Diese Tools helfen Ihnen dabei, das Nutzungsprofil des GuardDuty Security Agents für Ihre Anwendungen zu ermitteln. Anschließend können Sie die Größe Ihrer Fargate-Aufgabe bei Bedarf anpassen, um mögliche Speicherausfälle zu vermeiden.

Überwachen Sie Amazon ECS-Container mit ECS Exec

Mit Amazon ECS Exec können Sie direkt mit Containern interagieren, ohne zuerst mit dem Host-Container-Betriebssystem interagieren, eingehende Ports öffnen oder SSH-Schlüssel verwalten zu müssen. Sie können ECS Exec verwenden, um Befehle in einem Container auszuführen oder eine Shell zu erhalten, der auf einer Amazon-EC2-Instance oder auf AWS Fargate ausgeführt wird. Dies erleichtert das Sammeln von Diagnoseinformationen und die schnelle Fehlerbehebung. Beispielsweise können Sie ECS Exec in einem Entwicklungsumfeld verwenden, um problemlos mit verschiedenen Prozessen in Ihren Containern zu interagieren und Ihre Anwendungen zu beheben. Und in Produktionsszenarien können Sie sich damit glasklaren Zugriff auf Ihre Container verschaffen, um Probleme zu debuggen.

Sie können Befehle in einem laufenden Linux- oder Windows-Container mit ECS Exec über die Amazon ECS-API, AWS Command Line Interface (AWS CLI), AWS SDKs oder die AWS Copilot-CLI ausführen. Einzelheiten zur Verwendung von ECS Exec sowie eine Videoanleitung mit der AWS Copilot-CLI finden Sie in der Copilot-Dokumentation. GitHub

Sie können ECS Exec auch verwenden, um strengere Richtlinien für die Zugriffskontrolle einzuhalten. Wenn Sie dieses Feature selektiv aktivieren, können Sie steuern, wer Befehle ausführen kann und für welche Tasks diese Befehle ausgeführt werden können. Mit einem Protokoll aller Befehle und ihrer Ausgabe können Sie mit ECS Exec anzeigen, welche Aufgaben ausgeführt wurden, und Sie können überprüfen, wer CloudTrail auf einen Container zugegriffen hat.

Überlegungen

Zu diesem Thema sollten Sie sich mit den folgenden Aspekten vertraut machen, die mit ECS Exec verbunden sind:

- ECS Exec wird derzeit nicht mit dem unterstützt. AWS Management Console
- ECS Exec wird für Aufgaben unterstützt, die auf der folgenden Infrastruktur ausgeführt werden:
  - Linux-Container auf Amazon EC2 auf jeder Amazon-ECS-optimierten AMI, einschließlich Bottlerocket
  - Linux- und Windows-Container auf externen Instances (Amazon ECS Anywhere)
  - Linux- und Windows-Container auf AWS Fargate
  - Windows-Container auf Amazon EC2 auf folgenden Amazon-ECS-optimierten Windows-AMIs (mit der Container-Agent-Version `1.56` oder höher):
    - Amazon-ECS-optimierter Windows Server 2022 Full AMI
    - Amazon-ECS-optimierter Windows Server 2022 Core AMI
    - Amazon-ECS-optimierter Windows Server 2019 Full AMI
    - Amazon-ECS-optimierter Windows Server 2019 Core AMI
    - Amazon-ECS-optimiertes Windows Server 20H2 Core AMI
- ECS Exec und Amazon VPC
  - Wenn Sie Amazon-VPC-Endpunkte als Schnittstelle mit Amazon ECS verwenden, müssen Sie die Schnittstelle Amazon-VPC-Endpunkte für Systems Manager Session Manager (`ssmmessages`) erstellen. Weitere Informationen zu Systems Manager Manager-VPC-Endpunkten finden Sie im Benutzerhandbuch unter Verwenden AWS PrivateLink zum Einrichten eines VPC-Endpoints für Session Manager.AWS Systems Manager
  - Wenn Sie die Schnittstelle Amazon VPC-Endpunkte mit Amazon ECS verwenden und AWS KMS key für die Verschlüsselung verwenden, müssen Sie die Schnittstelle Amazon-VPC-Endpunkt für AWS KMS key erstellen. Weitere Informationen finden Sie unter Verbinden mit AWS KMS key über einen VPC-Endpunkt im AWS Key Management Service -Entwicklerhandbuch.
  - Wenn Sie Aufgaben haben, die auf Amazon EC2 EC2-Instances ausgeführt werden, verwenden Sie den `awsvpc` Netzwerkmodus. Wenn Sie keinen Internetzugang haben (z. B. nicht für die Verwendung eines NAT-Gateways konfiguriert), müssen Sie die Schnittstelle (Amazon VPC-Endpoints) für den Systems Manager Session Manager (`ssmmessages`) erstellen. Weitere Informationen zu Überlegungen zum `awsvpc`-Netzwerkmodus finden Sie unter Überlegungen. Weitere Informationen zu Systems Manager Manager-VPC-Endpunkten finden

Sie im Benutzerhandbuch unter Verwenden AWS PrivateLink zum Einrichten eines VPC-Endpoints für Session Manager.AWS Systems Manager

- ECS Exec und SSM
  - Wenn ein Benutzer Befehle für einen Container mit ECS Exec ausführt, werden diese Befehle als der `root`-Benutzer ausgeführt. Der SSM Agent und seine untergeordneten Prozesse werden auch dann als Root ausgeführt, wenn Sie eine Benutzer-ID für den Container angeben.
  - Der SSM-Agent erfordert, dass in das Container-Dateisystem geschrieben werden kann, um die erforderlichen Verzeichnisse und Dateien zu erstellen. Daher wird das Schreibschützen der Root-Dateisystem mit dem `readonlyRootFilesystem`-Aufgabendefinitionsparameter oder eine andere Methode nicht unterstützt.
  - Obwohl das Starten von SSM-Sitzungen außerhalb der `execute-command`-Aktion möglich ist, führt dies dazu, dass die Sitzungen nicht protokolliert und gegen das Sitzungslimit gezählt werden. Wir empfehlen, diesen Zugriff einzuschränken, indem Sie die `ssm:start-session`-Aktion mit einer IAM-Richtlinie verweigern. Weitere Informationen finden Sie unter Beschränken des Zugriffs auf die Aktion „Sitzung starten“.
- Die folgenden Funktionen werden als Sidecar-Container ausgeführt. Daher müssen Sie den Namen des Containers angeben, für den der Befehl ausgeführt werden soll.
  - Laufzeit-Überwachung
  - Service Connect
- Benutzer können alle Befehle ausführen, die im Container-Kontext verfügbar sind. Die folgenden Aktionen können zu verwaisten und Zombie-Prozessen führen: Beenden des Hauptprozesses des Containers, Beenden des Befehls-Agenten und Löschen von Abhängigkeiten. Um Zombie-Prozesse zu bereinigen, empfehlen wir Ihnen, das `initProcessEnabled`-Flag an Ihre Aufgabendefinition anzufügen.
- ECS Exec verwendet etwas CPU und Arbeitsspeicher. Sie sollten dies berücksichtigen, wenn Sie die CPU- und Arbeitsspeicherressourcenzuordnungen in Ihrer Aufgabendefinition angeben.
- Sie müssen AWS CLI Version `1.22.3` oder höher oder AWS CLI Version `2.3.6` oder höher verwenden. Informationen zur Aktualisierung von finden Sie unter Installation oder Aktualisierung der neuesten Version von AWS CLI im AWS Command Line Interface Benutzerhandbuch Version 2. AWS CLI
- Sie können nur eine ECS-Exec-Sitzung pro Prozess-ID-Namespace (PID) haben. Wenn Sie einen PID-Namespace in einer Aufgabe freigeben, können Sie ECS-Exec-Sitzungen nur in einem Container starten.

- Für die ECS Exec-Sitzung ist eine Zeitüberschreitung von 20 Minuten konfiguriert. Dieser Wert kann nicht geändert werden.
- Sie können ECS Exec nicht für vorhandene Aufgaben aktivieren. Es kann nur für neue Aufgaben aktiviert werden.
- Sie können ECS Exec nicht verwenden`run-task`, wenn Sie eine Aufgabe auf einem Cluster starten, der verwaltete Skalierung mit asynchroner Platzierung verwendet (eine Aufgabe ohne Instanz starten).
- Sie können ECS Exec nicht für Microsoft Nano Server-Container ausführen.

Voraussetzungen

Bevor Sie mit der Verwendung von ECS Exec beginnen, stellen Sie sicher, dass Sie die folgenden Aktionen abgeschlossen haben:

- Installieren und Konfigurieren der AWS CLI. Weitere Informationen finden Sie unter AWS CLI.
- Installieren Sie das Session Manager-Plug-In für. AWS CLI Weitere Informationen finden Sie unter Installieren des Session Manager-Plugins für AWS CLI.
- Sie müssen eine Aufgabenrolle mit den entsprechenden Berechtigungen für ECS Exec verwenden. Weitere Informationen finden Sie unter Aufgaben-IAM-Rollen.
- ECS Exec hat Versionsanforderungen abhängig davon, ob Ihre Aufgaben auf Amazon EC2 oder AWS Fargate gehostet werden:
  - Wenn Sie Amazon EC2 verwenden, müssen Sie ein für Amazon ECS optimiertes AMI verwenden, das nach dem 20. Januar 2021 mit einer Agent-Version von 1.50.2 oder höher veröffentlicht wurde. Weitere Informationen finden Sie unter Amazon-ECS-optimierte AMIs.
  - Wenn Sie verwenden AWS Fargate, müssen Sie die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` (Windows) verwenden. Weitere Informationen finden Sie unter AWS Fargate - Plattformversionen.

Architektur

ECS Exec verwendet den AWS Systems Manager (SSM) Session Manager, um eine Verbindung mit dem laufenden Container herzustellen, und verwendet AWS Identity and Access Management (IAM-) Richtlinien, um den Zugriff auf laufende Befehle in einem laufenden Container zu steuern. Dies wird ermöglicht, indem die notwendigen SSM Agent-Binärdateien in den Container eingebunden

werden. Der Amazon ECS oder AWS Fargate -Agent ist dafür verantwortlich, den SSM-Core-Agenten im Container zusammen mit Ihrem Anwendungscode zu starten. Weitere Informationen erhalten Sie unter Systems Manager Session Manager.

Sie können anhand des `ExecuteCommand` Event in überprüfen, welcher Benutzer auf den Container zugegriffen hat, AWS CloudTrail und jeden Befehl (und seine Ausgabe) in Amazon S3 oder Amazon CloudWatch Logs protokollieren. Um Daten zwischen dem lokalen Client und dem Container mit Ihrem eigenen Verschlüsselungsschlüssel zu verschlüsseln, müssen Sie den Schlüssel AWS Key Management Service (AWS KMS) angeben.

Verwenden von ECS Exec

Änderungen an optionalen Aufgabendefinition

Wenn Sie den Aufgabendefinitionsparameter `initProcessEnabled` auf setzen`true`, wird der Init-Prozess innerhalb des Containers gestartet. Dadurch werden alle gefundenen untergeordneten Prozesse des Zombie-SSM-Agents entfernt. Folgendes ist ein Beispiel.

```
{
    "taskRoleArn": "ecsTaskRole",
    "networkMode": "awsvpc",
    "requiresCompatibilities": [
        "EC2",
        "FARGATE"
    ],
    "executionRoleArn": "ecsTaskExecutionRole",
    "memory": ".5 gb",
    "cpu": ".25 vcpu",
    "containerDefinitions": [
        {
            "name": "amazon-linux",
            "image": "amazonlinux:latest",
            "essential": true,
            "command": ["sleep","3600"],
            "linuxParameters": {
                "initProcessEnabled": true
            }
        }
    ],
    "family": "ecs-exec-task"
```

```
}
```

Aktivieren von ECS Exec für Ihre Aufgaben und Services

Sie können die ECS Exec-Funktion für Ihre Dienste und eigenständigen Aufgaben aktivieren, indem Sie das `--enable-execute-command` Flag angeben, wenn Sie einen der folgenden AWS CLI Befehle verwenden: `create-service`, `update-servicestart-task`, oder. `run-task`

Wenn Sie beispielsweise den folgenden Befehl ausführen, wird die ECS Exec-Funktion für einen neu erstellten Dienst aktiviert, der auf Fargate ausgeführt wird. Weitere Informationen zum Erstellen von Services finden Sie unter create-service.

```
aws ecs create-service \
    --cluster cluster-name \
    --task-definition task-definition-name \
    --enable-execute-command \
    --service-name service-name \
    --launch-type FARGATE \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
 \
    --desired-count 1
```

Nachdem Sie ECS Exec für eine Aufgabe aktiviert haben, können Sie den folgenden Befehl ausführen, um zu bestätigen, dass die Aufgabe für die Verwendung bereit ist. Wenn die `lastStatus`-Eigenschaft des `ExecuteCommandAgent` als `RUNNING` aufgelistet wird und die `enableExecuteCommand`-Eigenschaft auf `true` festgelegt ist, dann ist Ihre Aufgabe fertig.

```
aws ecs describe-tasks \
    --cluster cluster-name \
    --tasks task-id
```

Das folgende Ausgabeschnippsel ist ein Beispiel davon, was Sie sehen könnten.

```
{
    "tasks": [
        {
            ...
            "containers": [
                {
```

```
                    ...
                    "managedAgents": [
                        {
                            "lastStartedAt": "2021-03-01T14:49:44.574000-06:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                    ]
                }
            ],
            ...
            "enableExecuteCommand": true,
            ...
        }
    ]
}
```

Ausführen von Befehlen mit ECS Exec

Nachdem Sie bestätigt haben, dass `ExecuteCommandAgent` ausgeführt wird, können Sie eine interaktive Shell in Ihrem Container mit dem folgenden Befehl öffnen. Wenn Ihre Aufgabe mehrere Container enthält, müssen Sie den Containernamen mithilfe des `--container`-Flag angeben. Amazon ECS unterstützt nur das Initiieren interaktiver Sitzungen. Daher müssen Sie das `--interactive`-Flag verwenden.

Mit dem folgenden Befehl wird ein interaktiver /bin/sh Befehl für einen Container namens container-name für eine Aufgabe mit der ID task-id ausgeführt.

Die *Task-ID* ist der Amazon-Ressourcenname (ARN) der Aufgabe.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Protokollierung mit ECS Exec

Einschalten der Protokollierung Ihrer Aufgaben und Dienste

> **Important**
>
> Weitere Informationen zur CloudWatch Preisgestaltung finden Sie unter CloudWatch Preise. Amazon ECS stellt außerdem Überwachungsmetriken bereit, die ohne zusätzliche Gebühren bereitgestellt werden. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS mit CloudWatch .

Amazon ECS bietet eine Standardkonfiguration für die Protokollierung von Befehlen, die mit ECS Exec ausgeführt werden, indem CloudWatch Protokolle mithilfe des `awslogs` Protokolltreibers, der in Ihrer Aufgabendefinition konfiguriert ist, an Logs gesendet werden. Wenn Sie eine benutzerdefinierte Konfiguration bereitstellen möchten, unterstützt AWS CLI ein `--configuration`-Flag sowohl für die `create-cluster`- und `update-cluster`-Befehle. Es ist auch wichtig zu wissen, dass das Container-Image installiert sein `cat` muss`script`, damit die Befehlsprotokolle korrekt in Amazon S3 oder CloudWatch Logs hochgeladen werden können. Weitere Informationen zum Erstellen von Clustern finden Sie unter create-cluster.

> **Note**
>
> Diese Konfiguration verarbeitet nur die Protokollierung der `execute-command`-Sitzung. Dies wirkt sich nicht auf die Protokollierung Ihrer Anwendung aus.

Das folgende Beispiel erstellt einen Cluster und protokolliert dann die Ausgabe in Ihren CloudWatch Logs LogGroup named `cloudwatch-log-group-name` und Ihrem Amazon S3 S3-Bucket named`s3-bucket-name`.

Sie müssen einen vom AWS KMS Kunden verwalteten Schlüssel verwenden, um die Protokollgruppe zu verschlüsseln, wenn Sie die `CloudWatchEncryptionEnabled` Option auf `true` setzen. Informationen zum Verschlüsseln der Protokollgruppe finden Sie unter Verschlüsseln von Protokolldaten im Abschnitt CloudWatch Protokolle mithilfe von AWS Key Management Service Protokollen im Amazon CloudWatch Logs Benutzerhandbuch.

```
aws ecs create-cluster \
```

```
    --cluster-name cluster-name \
    --configuration executeCommandConfiguration="{ \
        kmsKeyId=string, \
        logging=OVERRIDE, \
        logConfiguration={ \
            cloudWatchLogGroupName=cloudwatch-log-group-name, \
            cloudWatchEncryptionEnabled=true, \
            s3BucketName=s3-bucket-name, \
            s3EncryptionEnabled=true, \
            s3KeyPrefix=demo \
        } \
    }"
```

Die `logging`-Eigenschaft bestimmt das Verhalten der Protokollierungsfunktion von ECS Exec:

- `NONE`: Die Protokollierung ist ausgeschaltet.
- `DEFAULT`: Protokolle werden an den konfigurierten `awslogs` Treiber gesendet. Wenn der Treiber nicht konfiguriert ist, wird kein Protokoll gespeichert.
- `OVERRIDE`: Protokolle werden an die bereitgestellten Amazon CloudWatch Logs- LogGroup, Amazon S3 S3-Bucket oder beide gesendet.

Für Amazon CloudWatch Logs oder Amazon S3 Logging sind IAM-Berechtigungen erforderlich

Um die Protokollierung zu aktivieren, benötigt die Amazon-ECS-Aufgabenrolle, die in Ihrer Aufgabendefinition referenziert wird, zusätzliche Berechtigungen. Diese zusätzlichen Berechtigungen können der Aufgabenrolle als Richtlinie hinzugefügt werden. Sie unterscheiden sich, je nachdem, ob Sie Ihre Protokolle an Amazon CloudWatch Logs oder Amazon S3 weiterleiten.

Amazon CloudWatch Logs

Die folgende Beispielrichtlinie fügt die erforderlichen Amazon CloudWatch Logs-Berechtigungen hinzu.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:region:account-id:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
        }
   ]
}
```

Amazon S3

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Amazon-S3-Berechtigungen hinzu.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
           "Effect": "Allow",
           "Action": [
               "s3:GetEncryptionConfiguration"
           ],
           "Resource": "arn:aws:s3:::s3-bucket-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::s3-bucket-name/*"
```

```
        }
     ]
    }
```

Für die Verschlüsselung mit Ihrem eigenen AWS KMS key (KMS-Schlüssel) sind IAM-Berechtigungen erforderlich

Standardmäßig verwenden die zwischen Ihrem lokalen Client und dem Container übertragenen Daten die TLS 1.2-Verschlüsselung, die dies AWS ermöglicht. Um Daten mit Ihrem eigenen KMS-Schlüssel weiter zu verschlüsseln, müssen Sie einen KMS-Schlüssel erstellen und die `kms:Decrypt`-Berechtigung für Ihre Aufgaben-IAM-Rolle hinzufügen. Diese Berechtigung wird von Ihrem Container verwendet, um die Daten zu entschlüsseln. Weitere Informationen zum Erstellen eines KMS-Schlüssels finden Sie unter Erstellen von Schlüsseln.

Sie fügen Ihrer Task-IAM-Rolle, für die die AWS KMS Berechtigungen erforderlich sind, die folgende Inline-Richtlinie hinzu. Weitere Informationen finden Sie unter ECS Exec-Berechtigungen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "kms-key-arn"
        }
    ]
}
```

Damit die Daten mit Ihrem eigenen KMS-Schlüssel verschlüsselt werden, muss dem Benutzer oder der Gruppe, der/die die `execute-command`-Aktion verwendet, die `kms:GenerateDataKey`-Berechtigung gewährt werden.

Die folgende Beispielrichtlinie für Ihren Benutzer oder Ihre Gruppe enthält die erforderliche Berechtigung, Ihren eigenen KMS-Schlüssel zu verwenden. Sie müssen den Amazon-Ressourcennamen (ARN) Ihres KMS-Schlüssels angeben.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "kms-key-arn"
        }
    ]
}
```

Verwenden von IAM-Richtlinien, um den Zugriff auf ECS Exec zu beschränken

Sie beschränken den Benutzerzugriff auf die API-Aktion „Befehl ausführen“, indem Sie einen oder mehrere der folgenden IAM-Policy-Bedingungsschlüssel verwenden:

- `aws:ResourceTag/clusterTagKey`
- `ecs:ResourceTag/clusterTagKey`
- `aws:ResourceTag/taskTagKey`
- `ecs:ResourceTag/taskTagKey`
- `ecs:container-name`
- `ecs:cluster`
- `ecs:task`
- `ecs:enable-execute-command`

Mit dem folgenden Beispiel einer IAM-Richtlinie können Benutzer Befehle in Containern ausführen, die innerhalb von Aufgaben mit einem Tag ausgeführt werden, das einen `environment`-Schlüssel und `development`-Wert hat und in einem Cluster mit dem Namen `cluster-name`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
```

```
            ],
         "Resource": [
                "arn:aws:ecs:region:aws-account-id:task/cluster-name/*",
                "arn:aws:ecs:region:aws-account-id:cluster/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ecs:ResourceTag/environment": "development"
                }
            }
        }
    ]
}
```

Im folgenden IAM-Richtlinienbeispiel können Benutzer die `execute-command`-API nicht verwenden, wenn der Containername `production-app` lautet.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ecs:ExecuteCommand"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:container-name": "production-app"
                }
            }
        }
    ]
}
```

Mit der folgenden IAM-Richtlinie können Benutzer Aufgaben nur starten, wenn ECS Exec deaktiviert ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:enable-execute-command": "false"
                }
            }
        }
    ]
}
```

> **Note**
>
> Da die `execute-command`API-Aktion nur Task- und Clusterressourcen in einer Anforderung enthält, werden nur Cluster- und Task-Tags ausgewertet.

Weitere Informationen zu IAM-Richtlinienbedingungsschlüsseln finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für den Amazon Elastic Container Service in der Service Authorization-Referenz.

Beschränken des Zugriffs auf die Aktion „Sitzung starten“

Während das Starten von SSM-Sitzungen auf Ihrem Container außerhalb von ECS Exec möglich ist, kann dies möglicherweise dazu führen, dass die Sitzungen nicht protokolliert werden. Sitzungen, die außerhalb von ECS Exec gestartet wurden, zählen ebenfalls zum Sitzungskontingent. Wir empfehlen, diesen Zugriff einzuschränken, indem Sie die `ssm:start-session`-Aktion direkt für Ihre Amazon-ECS-Aufgaben mithilfe einer IAM-Richtlinie verweigern. Sie können den Zugriff auf alle Amazon-ECS-Aufgaben oder bestimmte Aufgaben basierend auf den verwendeten Tags verweigern.

Nachfolgend finden Sie eine Beispiel-IAM-Richtlinie, die den Zugriff auf die `ssm:start-session`-Aktion für Tasks in allen Regionen mit einem angegebenen Clusternamen verweigert. Optional können Sie einen Platzhalter mit *`cluster-name`* einschließen.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
                "arn:aws:ecs:region:aws-account-id:task/cluster-name/*",
                "arn:aws:ecs:region:aws-account-id:cluster/*"
            ]
        }
    ]
}
```

Nachfolgend finden Sie eine Beispiel-IAM-Richtlinie, die den Zugriff auf die `ssm:start-session`-Aktion für Ressourcen in allen Regionen, die mit Tag-Schlüssel `Task-Tag-Key` markiert sind und Tag-Wert `Exec-Task` haben, verweigert.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ecs:*:*:task/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Task-Tag-Key": "Exec-Task"
                }
            }
        }
    ]
}
```

Hilfe zu Problemen, auf die Sie bei der Verwendung von Amazon ECS Exec stoßen könnten, finden Sie unter Problembehandlung mit Exec.

AWS Compute Optimizer Empfehlungen für Amazon ECS

AWS Compute Optimizer generiert Empfehlungen für Amazon ECS-Aufgaben- und Containergrößen. Weitere Informationen finden Sie unter Was ist AWS Compute Optimizer? im AWS Compute Optimizer -Benutzerhandbuch.

Empfehlungen zur Aufgaben- und Containergröße für Amazon ECS-Services auf AWS Fargate

AWS Compute Optimizer generiert Empfehlungen für Amazon ECS-Services auf AWS Fargate. AWS Compute Optimizer empfiehlt die Größe der Task-CPU und des Task-Speichers sowie die Größe der Container-CPU, des Container-Speichers und der Reservierungsgröße des Container-Speichers. Diese Empfehlungen werden auf den folgenden Seiten der Compute-Optimizer-Konsole angezeigt.

- Empfehlungen für Amazon-ECS-Services auf Fargate-Seite
- Amazon-ECS-Services auf der Fargate-Detail-Seite

Weitere Informationen finden Sie im AWS Compute Optimizer -Benutzerhandbuch unter Empfehlungen für Amazon-ECS-Services auf Fargate anzeigen.

Amazon-ECS-Fehlersuche

Möglicherweise müssen Sie Probleme mit Ihren Load Balancern, Aufgaben, Diensten oder Container-Instances beheben. In diesem Kapitel finden Sie Diagnoseinformationen für den Amazon-ECS-Container-Agenten, den Docker-Daemon auf der Container-Instance und das Service-Ereignisprotokoll in der Amazon-ECS-Konsole.

Informationen zu gestoppten Aufgaben finden Sie in einer der folgenden Seiten.

Aktion	Weitere Informationen
Beheben Sie Fehler bei beendeten Aufgaben.	Fehler beim Beenden von Amazon ECS-Aufgaben anzeigen
Fehler bei beendeten Aufgaben anzeigen.	Fehler beheben, bei denen Amazon ECS Aufgaben beendet hat
Überprüfen Sie die Fehlercodes für abgebrochene Aufgaben.	Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS
Überprüfen Sie die Fehler bei CannotPullContainer Aufgaben.	CannotPullContainer Aufgabenfehler in Amazon ECS
IAM-Rollenanfragen für Aufgaben anzeigen	IAM-Rollenanfragen für Amazon ECS-Aufgaben anzeigen

Informationen zu Dienstfehlern finden Sie in einer der folgenden Kategorien.

Aktion	Weitere Informationen	
Meldungen zu Dienstereignissen anzeigen.	Ereignismeldungen des Amazon ECS-Service anzeigen	
Überprüfen Sie die Meldungen zu Serviceereignissen.	Amazon ECS-Serviceereignismeldungen	
Überprüfen Sie die Probleme mit dem Load Balancer.	Fehlerbehebung bei Service Load Balancers in Amazon ECS	
Überprüfen Sie die Probleme mit der auto Skalierung von Diensten.	Fehlerbehebung bei Service Auto Scaling in Amazon ECS	

Informationen zu Fehlern bei der Aufgabendefinition finden Sie in einem der folgenden Artikel.

Aktion	Weitere Informationen	
Beheben Sie den Speicherfehler bei der Aufgabendefinition.	Problembehandlung bei ungültigen CPU- oder Speicherfehlern mit Amazon ECS-Aufgabendefinition	

Informationen zu Amazon ECS-Agentenfehlern finden Sie in einer der folgenden Seiten.

Aktion	Weitere Informationen	
Amazon ECS-Container-Agent-Protokolle anzeigen.	Amazon ECS-Container-Agent-Protokolle anzeigen	
Erfahren Sie, wie Sie Amazon ECS-Protokolle sammeln.	Sammeln von Container-Protokollen mit Amazon ECS Logs Collector	

Aktion	Weitere Informationen	
Rufen Sie Diagnosedetails mit dem Amazon ECS-Agenten ab.	Rufen Sie Amazon ECS-Diagnosedetails mit Agenten-Introspektion ab	

Informationen zu Docker-Fehlern finden Sie in einem der folgenden Artikel.

Aktion	Weitere Informationen	
Verwenden Sie die Docker-Diagnose.	Docker-Diagnose in Amazon ECS	
Schalten Sie den Docker-Debug-Modus ein.	Konfiguration der ausführlichen Ausgabe vom Docker-Daemon in Amazon ECS	
Beheben Sie den Docker-API-Fehler 500.	Fehlerbehebung beim Docker `API error (500): devmapper` in Amazon ECS	

Informationen zu ECS Exec- und Amazon ECS Anywhere Anywhere-Fehlern finden Sie in einer der folgenden Kategorien.

Aktion	Weitere Informationen	
Beheben Sie Fehler bei ECS Exec.	Probleme mit Amazon ECS Exec beheben	
Problembehandlung bei Amazon ECS Anywhere.	Probleme mit Amazon ECS Anywhere beheben	

Informationen zu Drosselungsproblemen finden Sie in einer der folgenden Seiten.

Aktion	Weitere Informationen
Erfahren Sie mehr über die Drosselungsquoten von Fargate.	AWS Fargate Drosselung der Kontingente
Lernen Sie die Best Practices für Amazon ECS-Throttling kennen.	Probleme mit der Drosselung von Amazon ECS lösen

Informationen zu API-Fehlern finden Sie in einem der folgenden Artikel.

Aktion	Weitere Informationen
Beheben Sie API-Fehler.	Gründe für den Ausfall der Amazon ECS-API

Fehler beheben, bei denen Amazon ECS Aufgaben beendet hat

Wenn Ihre Aufgabe nicht gestartet werden kann, wird in der Konsole und in den `describe-tasks` Ausgabeparametern (stoppedReason und) eine Fehlermeldung angezeigt. StoppedCode Die folgenden Abschnitte enthalten zusätzliche Informationen zur Behebung von Problemen mit gestoppten Aufgaben.

Die folgenden Seiten enthalten Informationen zu gestoppten Aufgaben.

- Erfahren Sie mehr über Änderungen an Fehlermeldungen zu beendeten Aufgaben.

  Amazon ECS hat die Aktualisierung von Task-Fehlermeldungen gestoppt
- Sehen Sie sich Ihre angehaltenen Aufgaben an, um Informationen über die Ursache zu erhalten.

  Fehler beim Beenden von Amazon ECS-Aufgaben anzeigen
- Erfahren Sie mehr über die Fehlermeldungen zu beendeten Aufgaben und mögliche Gründe für die Fehler.

  Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS

- Erfahren Sie, wie Sie die Konnektivität gestoppter Aufgaben überprüfen und die Fehler beheben können.

  Überprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat

Amazon ECS hat die Aktualisierung von Task-Fehlermeldungen gestoppt

Ab dem 14. Juni 2024 ändert das Amazon ECS-Team die Fehlermeldungen für gestoppte Aufgaben wie in den folgenden Tabellen beschrieben. Das `stopCode` wird sich nicht ändern. Wenn Ihre Anwendungen von exakten Fehlermeldungszeichenfolgen abhängen, müssen Sie Ihre Anwendungen mit den neuen Zeichenfolgen aktualisieren. Wenn Sie Hilfe bei Fragen oder Problemen benötigen, wenden Sie sich an AWS Support.

> **Note**
>
> Wir empfehlen Ihnen, sich bei Ihrer Automatisierung nicht auf die Fehlermeldungen zu verlassen, da sich die Fehlermeldungen ändern können.

CannotPullContainerError

Alte Fehlermeldung	Neue Fehlermeldung	
CannotPullContainerError*: Fehlerantwort vom Daemon: Pull-Zugriff für das Repository verweigert, das Repository existiert nicht oder erfordert möglicherweise eine Docker-Anmeldung: verweigert: Benutzer: roleARN*	CannotPullContainerError: Die Aufgabe kann das Bild nicht abrufen. Vergewissern Sie sich, dass die Rolle berechtigt ist, Bilder aus der Registrierung abzurufen. Fehlerantwort vom Daemon: Pull-Zugriff für das *Repository* verweigert, das Repository ist nicht vorhanden oder erfordert möglicherweise eine Docker-Anmeldung: verweigert: Benutzer: *roleARN* ist nicht berechtigt,: ecr: BatchGetI	

Alte Fehlermeldung	Neue Fehlermeldung	
	mage on resource: *image* auszuführen, da keine identität sbasierte Richtlinie die Aktion ecr: zulässt. BatchGetImage	
	CannotPullContainerError: Die Aufgabe kann das Bild nicht abrufen. Prüfen Sie, ob das Bild existiert. Fehlerantwort vom Daemon: Pull-Zugriff für das *Repository* verweiger t, das Repository existiert nicht oder erfordert möglicherweise eine Docker-Anmeldung: verweigert: Der angeforde rte Zugriff auf die Ressource wurde verweigert.	
CannotPullContainerError: Fehlerantwort vom Daemon: *ImageURI abrufen: net/ http: Anfrage* beim Warten auf Verbindung abgebrochen	CannotPullContainerError: Die Aufgabe kann das Bild nicht abrufen. Überprüfen Sie Ihre Netzwerkkonfigurat ion. Fehlerantwort vom Daemon: *Bild* abrufen: net/ http: Anfrage beim Warten auf Verbindung abgebrochen (Client.Timeout wurde beim Warten auf Header überschri tten)	

ResourceNotFoundException

Alte Fehlermeldung	Neue Fehlermeldung	
Abrufen geheimer Daten aus AWS Secrets Manager der Region us-west-2: secret *SecretArn* :: Secrets Manager kann das ResourceNotFoundException angegebene Geheimnis nicht finden.	ResourceNotFoundException: Die Aufgabe kann das Geheimnis mit dem ARN '*sercRetarn' von nicht abrufen.* AWS Secrets Manager Prüfen Sie, ob das Geheimnis in der angegebenen Region existiert. ResourceNotFoundException: Geheime Daten aus AWS Secrets Manager einer *Region* abrufen: secret *SecretArn* : ResourceNotFoundException: Secrets Manager kann das angegebene Geheimnis nicht finden.	

Fehler beim Beenden von Amazon ECS-Aufgaben anzeigen

Wenn Sie Probleme beim Starten einer Aufgabe haben, wird Ihre Aufgabe möglicherweise aufgrund von Anwendungs- oder Konfigurationsfehlern angehalten. Sie führen beispielsweise eine Aufgabe aus, und die Aufgabe zeigt einen Status `PENDING` an, verschwindet dann aber.

Wenn Ihre Aufgabe von einem Amazon-ECS-Service erstellt wurde, werden die Aktionen, die Amazon ECS zur Wartung des Service durchführt, in den Service-Ereignissen veröffentlicht. Sie können die Ereignisse in den AWS Management Console,, AWS SDKs AWS CLI, der Amazon ECS-API oder in Tools anzeigen, die die SDKs und die API verwenden. Zu diesen Ereignissen gehört, dass Amazon ECS eine Aufgabe anhält und ersetzt, weil die Container in der Aufgabe nicht mehr ausgeführt werden oder zu viele Zustandsprüfungen von Elastic Load Balancing fehlgeschlagen sind.

Wenn Ihre Aufgabe auf einer Container-Instance auf Amazon EC2 oder externen Computern ausgeführt wurde, können Sie sich auch die Protokolle der Container-Laufzeit und des Amazon ECS-Agenten ansehen. Diese Protokolle befinden sich auf der Amazon EC2 EC2-Host-Instance oder dem

externen Computer. Weitere Informationen finden Sie unter Amazon ECS-Container-Agent-Protokolle anzeigen.

Verfahren

Console

AWS Management Console

Die folgenden Schritte können verwendet werden, um gestoppte Aufgaben mithilfe der neuen AWS Management Console Version auf Fehler zu überprüfen.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie auf der Seite Cluster : ***Name*** die Registerkarte Tasks (Aufgaben).
5. Konfigurieren Sie den Filter so, dass gestoppte Aufgaben angezeigt werden. Wählen Sie für Gewünschten Status filtern die Option Angehalten oder Beliebiger Status.

   Die Option Angehalten zeigt Ihre angehaltenen Aufgaben an und Beliebiger Status zeigt alle Ihre Aufgaben an.
6. Wählen Sie die zu untersuchende angehaltene Aufgabe aus.
7. Wählen Sie in der Zeile für Ihre gestoppte Aufgabe in der Spalte Letzter Status die Option Angehalten aus.

   In einem Popup-Fenster wird der Grund für das Abbrechen angezeigt.

AWS CLI

1. Rufen Sie eine Liste der gestoppten Aufgaben in einem Cluster ab. Die Ausgabe enthält den Amazon-Ressourcennamen (ARN) der Aufgabe, die Sie zur Beschreibung der Aufgabe benötigen.

```
aws ecs list-tasks \
     --cluster cluster_name \
     --desired-status STOPPED \
     --region region
```

2. Beschreiben Sie die angehaltene Aufgabe zum Abrufen der Informationen. Weitere Informationen finden Sie unter Describe-tasks in der AWS Command Line Interface Referenz.

```
aws ecs describe-tasks \
     --cluster cluster_name \
     --tasks arn:aws:ecs:region:account_id:task/cluster_name/task_ID \
     --region region
```

Verwenden Sie die folgenden Ausgabeparameter.

- `stopCode`- Der Stoppcode gibt beispielsweise an, warum eine Aufgabe gestoppt wurde ResourceInitializationError
- `StoppedReason`- Der Grund, warum die Aufgabe gestoppt wurde.
- `reason`(in der `containers` Struktur) — Geben Sie weitere Informationen zum angehaltenen Container an.

Nächste Schritte

Sehen Sie sich Ihre gestoppten Aufgaben an, um Informationen über die Ursache zu erhalten. Weitere Informationen finden Sie unter Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS.

Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS

Im Folgenden sind die möglichen Fehlermeldungen aufgeführt, die Sie möglicherweise erhalten, wenn Ihre Aufgabe unerwartet beendet wird.

Informationen dazu, wie Sie mithilfe von überprüfen können, ob bei Ihren gestoppten Tasks eine Fehlermeldung angezeigt wird AWS Management Console, finden Sie unterFehler beim Beenden von Amazon ECS-Aufgaben anzeigen.

Fehlercodes für abgebrochene Aufgaben sind mit einer Kategorie verknüpft, zum Beispiel "ResourceInitializationFehler“. Weitere Informationen zu den einzelnen Kategorien finden Sie im Folgenden:

Kategorie	Weitere Informationen
TaskFailedToStart	Behebung von Amazon TaskFailedToStart ECS-Fehlern
ResourceInitializationFehler	Behebung von Amazon ResourceInitializationError ECS-Fehlern
ResourceNotFoundException	Behebung von Amazon ResourceNotFoundException ECS-Fehlern
SpotInterruptionFehler	Behebung von Amazon SpotInterruption ECS-Fehlern
InternalError	Behebung von Amazon InternalError ECS-Fehlern
OutOfMemoryError	Behebung von Amazon OutOfMemoryError ECS-Fehlern
ContainerRuntimeFehler	Behebung von Amazon ContainerRuntimeError ECS-Fehlern
ContainerRuntimeTimeoutError	Behebung von Amazon ContainerRuntimeTimeoutError ECS-Fehlern
CannotStartContainerError	Behebung von Amazon CannotStartContainerError ECS-Fehlern
CannotStopContainerError	Behebung von Amazon CannotStopContainerError ECS-Fehlern

Kategorie	Weitere Informationen	
CannotInspectContainerError	Behebung von Amazon CannotInspectContainerError ECS-Fehlern	
CannotCreateVolumeError	Behebung von Amazon CannotCreateVolumeError ECS-Fehlern	
CannotPullBehälter	CannotPullContainer Aufgabenfehler in Amazon ECS	

Behebung von Amazon TaskFailedToStart ECS-Fehlern

Im Folgenden finden Sie einige `TaskFailedToStart` Fehlermeldungen und Maßnahmen, die Sie ergreifen können, um die Fehler zu beheben.

Unerwarteter EC2-Fehler beim Versuch, eine Netzwerkschnittstelle mit aktivierter öffentlicher IP-Zuweisung in Subnetz-ID zu erstellen

Dies passiert, wenn eine Fargate-Aufgabe, die den `awsvpc` Netzwerkmodus verwendet und in einem Subnetz mit einer öffentlichen IP-Adresse ausgeführt wird, und das Subnetz nicht über genügend IP-Adressen verfügt.

Die Anzahl der verfügbaren IP-Adressen finden Sie auf der Seite mit den Subnetzdetails in der Amazon EC2 EC2-Konsole oder über. `describe-subnets` Weitere Informationen finden Sie unter Ihr Subnetz anzeigen im Amazon VPC-Benutzerhandbuch.

Um dieses Problem zu beheben, können Sie ein neues Subnetz erstellen, in dem Sie Ihre Aufgabe ausführen können.

InternalError: <reason>

Dieser Fehler tritt auf, wenn eine ENI-Anlage angefordert wird. Amazon EC2 übernimmt asynchron die Bereitstellung der ENI. Der Bereitstellungsprozess nimmt Zeit in Anspruch. Amazon ECS hat ein Timeout für den Fall, dass es zu langen Wartezeiten oder nicht gemeldeten Fehlern kommt. Es gibt Zeiten, in denen die ENI bereitgestellt wird, aber der Bericht wird nach dem Fehler-Timeout an

Amazon ECS gesendet. In diesem Fall erkennt Amazon ECS den gemeldeten Aufgabenfehler mit einer verwendeten ENI.

Die ausgewählte Aufgabendefinition ist nicht mit der ausgewählten Rechenstrategie kompatibel

Dieser Fehler tritt auf, wenn Sie eine Aufgabendefinition mit einem Starttyp auswählen, der nicht dem Cluster-Kapazitätstyp entspricht. Weitere Informationen finden Sie unter Amazon-ECS-Starttypen. Sie müssen eine Aufgabendefinition auswählen, die dem Kapazitätsanbieter entspricht, der Ihrem Cluster zugewiesen ist.

Behebung von Amazon ResourceInitializationError ECS-Fehlern

Im Folgenden finden Sie einige `ResourceInitialization` Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

Geheimnisse oder Registrierungsauthentifizierung können nicht abgerufen werden: Die Aufgabe kann die Registrierungsauthentifizierung nicht aus Amazon ECR abrufen

Dieser Fehler tritt auf, wenn Ihre Aufgabe das in der Aufgabendefinition definierte Bild nicht abrufen kann.

Dieses Problem wird durch einen der folgenden Gründe verursacht:

Ursache des Fehlers..	Vorgehensweise	
Netzwerkverbindungsproblem zwischen dem Amazon ECR VPC-Endpunkt und der Aufgabe. Das Problem ist ein Netzwerkproblem, wenn Sie eine der folgenden Zeichenfolgen in der Fehlermeldung sehen: • Wählen Sie TCP • Wählen Sie UDP • <ip>:<port>: I/O-Timeout • net/http: TLS-Handshake-Timeout	Überprüfen Sie die Konnektivität zwischen der Aufgabe und dem Amazon VPC-Endpunkt:Überprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat.	

Ursache des Fehlers..	Vorgehensweise	
• gelesen: Verbindungs-Timeout • Beim Warten auf Header wurde der Client.Timeout überschritten • net/http: Die Anfrage wurde beim Warten auf die Verbindung abgebrochen • Signal: getötet • Frist für den Kontext überschritten		
Die in der Aufgabendefinition definierte Rolle hat nicht die Berechtigungen für Amazon ECR.	Fügen Sie der Aufgabena usführungsrolle die erforderl ichen Berechtigungen hinzu. Die Aufgabe verwendet eine der folgenden Rollen: • Für Aufgaben mit dem Starttyp Fargate ist dies die Aufgabenausführungsrolle. Weitere Informationen finden Sie unter Fargate-Aufgaben: Abrufen von Amazon ECR-Images über die Berechtigungen von Schnittstellen-Endpunkten. • Für Aufgaben mit dem EC2-Starttyp ist dies die Container-Instance-Rolle. Weitere Informationen finden Sie unter Amazon-ECR-Berechtigungen.	

Ursache des Fehlers..	Vorgehensweise	
Das Bild ARN existiert nicht	Sehen Sie sich das Bild an und überprüfen Sie dann Folgendes: Informationen zum Anzeigen Ihrer Bilder finden Sie unter Bilddetails in Amazon ECR anzeigen im Amazon Elastic Container Registry-Benutzerhandbuch. • Das Bild befindet sich in derselben Region wie die Aufgabe. Schieben Sie das Bild in die richtige Region. Aktualisieren Sie dann die Aufgabe mit dem neuen Image-ARN. Informationen zum Übertragen eines Bilds finden Sie unter Ein Bild in ein Amazon ECR-Repository übertragen im Amazon ECR-Benutzerhandbuch. Informationen zur Aktualisierung der Aufgabendefinition finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic	

Ursache des Fehlers..	Vorgehensweise	
	Container Service API-Referenz. • Die Aufgabendefinition hat den falschen Bild-ARN. Aktualisieren Sie die Aufgabendefinition. Informationen zur Aktualisierung der Aufgabendefinition finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.	

kann keine Geheimnisse oder Registrierungsauthentifizierung abrufen: Geheimnisse können nicht von SSM abgerufen werden: Die Aufgabe kann den geheimen '***SecretName***' nicht aus Systems Manager abrufen

Dieser Fehler tritt auf, wenn Ihre Aufgabe das in der Aufgabendefinition definierte Bild nicht mithilfe der Anmeldeinformationen in Systems Manager abrufen kann.

Dieses Problem wird durch einen der folgenden Gründe verursacht:

Ursache des Fehlers..	Vorgehensweise	
Netzwerkverbindungsproblem zwischen dem Systems Manager VPC-Endpunkt und der Aufgabe. Das Problem ist ein Netzwerkproblem, wenn Sie eine der	Überprüfen Sie die Konnektivität zwischen der Aufgabe und dem Systems Manager Manager-Endpunkt:Überprüfung, ob Amazon ECS die	

Ursache des Fehlers..	Vorgehensweise	
folgenden Zeichenfolgen in der Fehlermeldung sehen: • Wählen Sie TCP • Wählen Sie UDP • <ip>:<port>: I/O-Timeout • net/http: TLS-Handshake-Timeout • gelesen: Verbindungs-Timeout • Beim Warten auf Header wurde der Client.Timeout überschritten • net/http: Die Anfrage wurde beim Warten auf die Verbindung abgebrochen • Signal: getötet • Frist für den Kontext überschritten	Aufgabenkonnektivität beendet hat.	
Die in der Aufgabendefinition definierte Rolle hat nicht die Berechtigungen für Secrets Manager.	Fügen Sie der Aufgabena usführungsrolle die erforderl ichen Systems Manager Manager-Berechtigungen hinzu. Weitere Informationen finden Sie unter Secrets Manager- oder Systems Manager Manager-Berechtigu ngen.	

Ursache des Fehlers..	Vorgehensweise	
Der geheime ARN existiert nicht	Prüfen Sie, ob der ARN existiert. Weitere Informati onen finden Sie unter Suchen nach Systems Manager Manager-Parametern im AWS Systems Manager Benutzerh andbuch.	

kann keine Geheimnisse oder Registrierungsauthentifizierung abrufen: Geheimnisse können nicht von ASM abgerufen werden: Die Aufgabe kann das geheime '***SecreRn***' nicht aus Secrets Manager abrufen

Dieser Fehler tritt auf, wenn Ihre Fargate-Aufgabe das in der Aufgabendefinition definierte Bild nicht mit den Anmeldeinformationen in Secrets Manager abrufen kann.

Dieses Problem wird durch einen der folgenden Gründe verursacht:

Ursache des Fehlers..	Vorgehensweise	
Netzwerkverbindungsproblem zwischen dem Secrets Manager VPC-Endpunkt und der Aufgabe. Das Problem ist ein Netzwerkp roblem, wenn Sie eine der folgenden Zeichenfolgen in der Fehlermeldung sehen: • Wählen Sie TCP • Wählen Sie UDP • <ip>:<port>: I/O-Timeout • net/http: TLS-Handshake-Timeout	Überprüfen Sie die Konnektiv ität zwischen der Aufgabe und dem Secrets Manager Manager-Endpunkt. Weitere Informationen finden Sie unter Überprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat.	

Ursache des Fehlers..	Vorgehensweise	
• gelesen: Verbindungs-Timeout • Beim Warten auf Header wurde der Client.Timeout überschritten • net/http: Die Anfrage wurde beim Warten auf die Verbindung abgebrochen • Signal: getötet • Frist für den Kontext überschritten		
Die in der Aufgabendefinition definierte Aufgabenausführungsrolle hat nicht die Berechtigungen für Secrets Manager.	Fügen Sie der Aufgabenausführungsrolle die erforderlichen Berechtigungen für Secrets Manager hinzu. Weitere Informationen finden Sie unter Secrets Manager- oder Systems Manager Manager-Berechtigungen.	
Der geheime ARN existiert nicht	Überprüfen Sie, ob der ARN in Secrets Manager vorhanden ist. Informationen zum Anzeigen Ihrer Bilder finden Sie unter Find Secrets in Secrets Manager im Secrets Manager Developer Guide.	

Geheimnisse oder Registrierungsauthentifizierung können nicht abgerufen werden: Die Aufgabe kann das geheime '***SecretRn***' nicht aus Secrets Manager abrufen

Dieser Fehler tritt auf, wenn Ihre Aufgabe das in der Aufgabendefinition definierte Bild nicht mit den Anmeldeinformationen in Secrets Manager abrufen kann.

Der Fehler weist darauf hin, dass ein Netzwerkverbindungsproblem zwischen dem Systems Manager VPC-Endpunkt und der Aufgabe besteht.

Informationen zur Überprüfung der Konnektivität zwischen der Aufgabe und dem Endpunkt finden Sie unterÜberprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat.

Env-Dateien konnten nicht heruntergeladen werden: Die Aufgabe kann die Umgebungsvariablendateien nicht von Amazon S3 herunterladen

Dieser Fehler tritt auf, wenn Ihre Aufgabe Ihre Umgebungsdatei nicht von Amazon S3 herunterladen kann.

Ursache des Fehlers..	Vorgehensweise	
Problem mit der Netzwerkverbindung zwischen der Aufgabe und Amazon S3.	Überprüfen Sie die Konnektivität zwischen der Aufgabe und dem Amazon S3 S3-Endpunkt:Überprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat.	
Die in der Aufgabendefinition definierte Rolle hat nicht die Berechtigungen für Amazon S3.	Fügen Sie der Rolle die Amazon S3 S3-Berechtigung hinzu. Weitere Informationen finden Sie unter Amazon S3 S3-Dateispeicherberechtigungen.	

Logger-Argumente konnten nicht validiert werden: Die Aufgabe kann den in der Aufgabendefinition definierten CloudWatch ***Logs-Gruppennamen*** nicht finden. Es besteht ein Verbindungsproblem zwischen der Aufgabe und. CloudWatch

Dieser Fehler tritt auf, wenn Ihre Aufgabe die CloudWatch Protokollgruppe, die Sie in der Aufgabendefinition definiert haben, nicht findet.

Der Fehler weist darauf hin, dass die CloudWatch Gruppe in der Aufgabendefinition nicht existiert.

Sie können eine der folgenden Optionen ausführen, um dieses Problem zu beheben:

Um diese Option zu verwenden...	Vorgehensweise
Aktualisieren Sie die Aufgabendefinition, sodass die Konfiguration der Protokollgruppe in die Container-Definition aufgenommen wird.	Informationen zur Aktualisierung der Aufgabendefinition finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.
Erstellen Sie die Protokollgruppe in CloudWatch	a. Führen Sie den folgenden Befehl aus, um den Namen der Protokollgruppe abzurufen. `aws ecs describe-task-definition \ --task-definition task-def-name | jq -r.taskDefinition.containerDefinitions[].logConfiguration` b. Erstellen Sie die Protokollgruppe. Weitere Informationen finden Sie unter Erstellen einer Protokollgruppe in CloudWatch Logs im Amazon CloudWatch Logs-Benutzerhandbuch.

der Protokollierungstreiber konnte nicht initialisiert werden

Dieser Fehler tritt auf, wenn Ihre Aufgabe die CloudWatch Protokollgruppe, die Sie in der Aufgabendefinition definiert haben, nicht findet.

Der Fehler weist darauf hin, dass die CloudWatch Gruppe in der Aufgabendefinition nicht existiert.

Sie können eine der folgenden Optionen ausführen, um dieses Problem zu beheben:

Um diese Option zu verwenden...	Vorgehensweise	
Aktualisieren Sie die Aufgabendefinition, sodass die Konfiguration der Protokollgruppe in die Container-Definition aufgenommen wird.	Informationen zur Aktualisierung der Aufgabendefinition finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.	
Erstellen Sie die Protokollgruppe in CloudWatch	a. Führen Sie den folgenden Befehl aus, um den Namen der Protokollgruppe abzurufen. `aws ecs describe-task-definition \ --task-definition task-def-name | jq -r.taskDefinition.containerDefinitions[].logConfiguration` b. Erstellen Sie die Protokollgruppe. Weitere Informationen finden Sie unter Erstellen einer Protokoll	

Um diese Option zu verwenden...	Vorgehensweise	
	gruppe in CloudWatch Logs im Amazon CloudWatch Logs-Benutzerhandbuch.	

EFS utils-Befehle zum Einrichten von EFS-Volumes konnten nicht aufgerufen werden

Die folgenden Probleme könnten Sie daran hindern, Ihre Amazon EFS-Volumes auf Ihren Anfragen bereitzustellen:

- Das Amazon EFS-Dateisystem ist nicht richtig konfiguriert.
- Die Aufgabe verfügt nicht über die erforderlichen Berechtigungen.
- Es gibt Probleme im Zusammenhang mit Netzwerk- und VPC-Konfigurationen.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Warum kann ich meine Amazon EFS-Volumes nicht für meine AWS Fargate Aufgaben bereitstellen auf AWS re:POST.

Behebung von Amazon ResourceNotFoundException ECS-Fehlern

Im Folgenden finden Sie einige `ResourceNotFoundException` Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

Die Aufgabe kann das Geheimnis mit dem ARN '***sercRetarn' von nicht abrufen.*** AWS Secrets ManagerÜberprüfen Sie, ob das Geheimnis in der angegebenen Region existiert.

Dieser Fehler tritt auf, wenn die Aufgabe das Geheimnis nicht aus Secrets Manager abrufen kann. Das bedeutet, dass das in der Aufgabendefinition angegebene (und in der Fehlermeldung enthaltene) Geheimnis nicht in Secrets Manager existiert.

Die Region ist in der Fehlermeldung enthalten.

Abrufen geheimer Daten aus AWS Secrets Manager der Region *`Region`*: secret *`SercreTarn`*: ResourceNotFoundException: Secrets Manager kann das angegebene Geheimnis nicht finden.

Informationen zum Auffinden von Geheimnissen finden Sie unter Suchen von Geheimnissen AWS Secrets Manager im Benutzerhandbuch.AWS Secrets Manager

Verwenden Sie die folgende Tabelle, um den Fehler zu ermitteln und zu beheben.

Problem	Aktionen
Der geheime Schlüssel befindet sich in einer anderen Region als die Aufgabend efinition.	a. Erstellen Sie das Geheimnis in derselben Region wie die Aufgabe. Weitere Informationen finden Sie unter Einen AWS Secrets Manager geheimen Schlüssel erstellen. b. Aktualisieren Sie die Aufgabendefinition mit dem neuen geheimen Schlüssel . Weitere Informationen finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.
Die Aufgabendefinition hat den falschen geheimen ARN. Das richtige Geheimnis ist in Secrets Manager vorhanden.	Aktualisieren Sie die Aufgabendefinition mit dem richtigen Geheimnis. Weitere Informationen finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.
Das Geheimnis ist nicht mehr vorhanden.	a. Erstellen Sie das Geheimnis in derselben Region wie die Aufgabe. Weitere Informationen finden Sie unter Einen AWS

Problem	Aktionen	
	Secrets Manager geheimen Schlüssel erstellen. b. Aktualisieren Sie die Aufgabendefinition mit dem neuen geheimen Schlüssel . Weitere Informationen finden Sie unter Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole oder RegisterTaskDefinition in der Amazon Elastic Container Service API-Referenz.	

Behebung von Amazon SpotInterruption ECS-Fehlern

Der `SpotInterruption` Fehler hat unterschiedliche Gründe für die Starttypen Fargate und EC2.

Fargate Starttyp

Der `SpotInterruption` Fehler tritt auf, wenn keine Fargate-Spot-Kapazität vorhanden ist oder wenn Fargate Spot-Kapazität zurücknimmt.

Sie können Ihre Aufgaben in mehreren Availability Zones ausführen lassen, um mehr Kapazität zu gewährleisten.

EC2-Starttyp

Dieser Fehler tritt auf, wenn keine Spot-Instances verfügbar sind oder EC2 die Spot-Instance-Kapazität zurücknimmt.

Sie können Ihre Instances in mehreren Availability Zones laufen lassen, um mehr Kapazität zu ermöglichen.

Behebung von Amazon InternalError ECS-Fehlern

Gilt für: Fargate-Starttyp

Der `InternalError` Fehler, wenn der Agent auf einen unerwarteten internen Fehler stößt, der nicht mit der Laufzeit zusammenhängt.

Dieser Fehler tritt nur auf, wenn die Plattformversion `1.4` oder höher verwendet wird.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Wie behebe ich eine Amazon ECS-Aufgabe, die in einem ECS-Cluster nicht gestartet werden konnte, auf AWS re:POST.

Behebung von Amazon OutOfMemoryError ECS-Fehlern

Im Folgenden finden Sie einige OutOfMemoryError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

Der Container wurde aufgrund von Speicherbelegung zerstört

Dieser Fehler tritt auf, wenn ein Container aufgrund von Prozessen im Container beendet wird, die mehr Speicher belegen als in der Aufgabendefinition zugewiesen wurde.

Behebung von Amazon ContainerRuntimeError ECS-Fehlern

Im Folgenden finden Sie einige ContainerRuntimeError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

ContainerRuntimeFehler

Dieser Fehler tritt auf, wenn der Agent einen unerwarteten Fehler von `containerd` für einen laufzeitspezifischen Vorgang erhält. Dieser Fehler wird normalerweise durch einen internen Fehler im Agent oder in der `containerd`-Laufzeitumgebung verursacht.

Dieser Fehler tritt nur auf, wenn Sie die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` oder höher (Windows) verwenden.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Warum wurde meine Amazon ECS-Aufgabe auf AWS re:POST gestoppt.

Behebung von Amazon ContainerRuntimeTimeoutError ECS-Fehlern

Im Folgenden finden Sie einige ContainerRuntimeTimeoutError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

Es konnte nicht zur Ausführung übergegangen werden. Timeout nach 1 m Wartezeit oder Docker-Timeout-Fehler

Dieser Fehler tritt auf, wenn ein Container innerhalb des Timeout-Zeitraums weder in einen Status `RUNNING` oder `STOPPED` wechseln konnte. Der Grund und der Timeout-Wert werden in der Fehlermeldung angegeben.

Behebung von Amazon CannotStartContainerError ECS-Fehlern

Im Folgenden finden Sie einige CannotStartContainerError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

Der Container-Status konnte nicht abgerufen werden: <reason>

Dieser Fehler tritt auf, wenn ein Container nicht gestartet werden kann.

Behebung von Amazon CannotStopContainerError ECS-Fehlern

Im Folgenden finden Sie einige CannotStopContainerError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

CannotStopContainerError

Dieser Fehler tritt auf, wenn ein Container nicht angehalten werden kann.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Warum wurde meine Amazon ECS-Aufgabe auf AWS re:POST gestoppt.

Behebung von Amazon CannotInspectContainerError ECS-Fehlern

Im Folgenden finden Sie einige CannotInspectContainerError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

CannotInspectContainerError

Dieser Fehler tritt auf, wenn der Container-Agent den Container nicht über die Containerlaufzeit beschreiben kann.

Wenn Sie die Plattformversion `1.3` oder eine frühere Version verwenden, gibt der Amazon ECS-Agent den Grund von Docker zurück.

Wenn Sie die Plattformversion `1.4.0` oder höher (Linux) `1.0.0` oder höher (Windows) verwenden, gibt der Fargate-Agent den Grund von `containerd` zurück.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Warum wurde meine Amazon ECS-Aufgabe auf AWS re:POST gestoppt.

Behebung von Amazon CannotCreateVolumeError ECS-Fehlern

Im Folgenden finden Sie einige CannotCreateVolumeError Fehlermeldungen und Maßnahmen, mit denen Sie die Fehler beheben können.

CannotCreateVolumeError

Dieser Fehler tritt auf, wenn der Agent die in der Aufgabendefinition angegeben Volume-Bereitstellung nicht erstellen kann.

Dieser Fehler tritt nur auf, wenn Sie die Plattformversion `1.4.0` oder höher (Linux) oder `1.0.0` oder höher (Windows) verwenden.

Informationen zum Debuggen und Beheben dieses Problems finden Sie unter Warum wurde meine Amazon ECS-Aufgabe auf AWS re:POST gestoppt.

CannotPullContainer Aufgabenfehler in Amazon ECS

Die folgenden Fehler weisen darauf hin, dass die Aufgabe nicht gestartet werden konnte, weil Amazon ECS das angegebene Container-Image nicht abrufen kann.

> **Note**
>
> Die 1.4 Fargate Plattformversion schneidet lange Fehlermeldungen ab.

Fehler

- Die Aufgabe kann das Image nicht abrufen. Vergewissern Sie sich, dass die Rolle berechtigt ist, Bilder aus der Registrierung abzurufen
- Die Aufgabe kann das Bild nicht abrufen. Überprüfen Sie Ihre Netzwerkkonfiguration
- API-Fehler (500): Rufen Sie https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/ ab: net/http: Anfrage wurde abgebrochen, während auf eine Verbindung gewartet wird
- API-Fehler
- schreibe /var/lib/docker/tmp/ Blob111111111GetImage: kein Speicherplatz mehr auf dem Gerät
- FEHLER: toomanyrequests: Zu viele Anfragen oder Sie haben Ihr Pull-Rate-Limit erreicht.

- Fehlerantwort vom Daemon: URL abrufen: net/http: Anfrage beim Warten auf Verbindung abgebrochen
- ref pull wurde 1 mal wiederholt: konnte nicht kopiert werden:: Fehler beim Öffnen httpReaderSeeker: unerwarteter Statuscode
- Pull-Zugriff verweigert
- Pull-Befehl fehlgeschlagen: Panik: Laufzeitfehler: ungültige Speicheradresse oder Nullzeiger-Dereferenzierung
- Fehler beim Abrufen der Image-Konfiguration/Fehler beim Abrufen der Image-Konfiguration
- Kontext abgebrochen

Die Aufgabe kann das Image nicht abrufen. Vergewissern Sie sich, dass die Rolle berechtigt ist, Bilder aus der Registrierung abzurufen

Dieser Fehler weist darauf hin, dass die Aufgabe das in der Aufgabendefinition angegebene Bild aufgrund von Berechtigungsproblemen nicht abrufen kann. Die Fehlermeldung enthält zusätzliche Informationen, die das Bild oder die Rolle angeben, die das Problem verursacht haben.

„Fehlerantwort vom Daemon: Der Pull-Zugriff für das *`Repository`* wurde verweigert, ist nicht vorhanden oder erfordert möglicherweise 'Docker-Anmeldung': verweigert: Benutzer: *`roleARN`* ist nicht berechtigt,: ecr: BatchGetImage on resource: *`image`* auszuführen, da keine identitätsbasierte Richtlinie die ecr: -Aktion zulässt.“ BatchGetImage

So beheben Sie dieses Problem

1. *`Überprüfen Sie, ob das Bild im i-Repository vorhanden ist.`* Informationen zum Anzeigen Ihrer Bilder finden Sie unter Bilddetails in Amazon ECR anzeigen im Amazon Elastic Container Registry-Benutzerhandbuch.
2. Stellen Sie sicher, dass der *`Role-ARN`* über die richtigen Berechtigungen zum Abrufen des Bildes verfügt.

   Informationen zum Anzeigen und Ändern von Rollen finden Sie im Benutzerhandbuch unter Ändern einer Rolle.AWS Identity and Access Management

   Die Aufgabe verwendet eine der folgenden Rollen:

   - Für Aufgaben mit dem Starttyp Fargate ist dies die Aufgabenausführungsrolle. Informationen zu den zusätzlichen Berechtigungen für Amazon ECR finden Sie unterFargate-Aufgaben: Abrufen von Amazon ECR-Images über die Berechtigungen von Schnittstellen-Endpunkten.

- Für Aufgaben mit dem EC2-Starttyp ist dies die Container-Instance-Rolle. Informationen zu den zusätzlichen Berechtigungen für Amazon ECR finden Sie unterAmazon-ECR-Berechtigungen.

Die Aufgabe kann das Bild nicht abrufen. Überprüfen Sie Ihre Netzwerkkonfiguration

Dieser Fehler weist darauf hin, dass die Aufgabe keine Verbindung zu Amazon ECR herstellen kann.

Informationen zur Überprüfung und Lösung des Problems finden Sie unterÜberprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat.

API-Fehler (500): Rufen Sie https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/ ab: net/http: Anfrage wurde abgebrochen, während auf eine Verbindung gewartet wird

Dieser Fehler weist darauf hin, dass bei einer Verbindung das Zeitlimit überschritten wurde, da keine Route zum Internet existiert.

Um dieses Problem zu lösen, können Sie:

- Für Aufgabe in öffentlichen Subnetzen geben Sie beim Starten der Aufgabe ENABLED unter Auto-assign public IP (Öffentliche IP automatisch zuweisen) an. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen.
- Für Aufgaben in privaten Subnetzen geben Sie beim Starten der Aufgabe DISABLED (DEAKTIVIERT) für Auto-assign public IP (Öffentliche IP automatisch zuweisen) an und konfigurieren ein NAT-Gateway in Ihrer VPC, um Anforderungen an das Internet weiterzuleiten. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.

API-Fehler

Dieser Fehler weist auf ein Verbindungsproblem mit dem Amazon ECR-Endpunkt hin.

Informationen zur Behebung dieses Problems finden Sie auf der AWS Support Website unter How can I resolve the Amazon ECR error "CannotPullContainerError: API error“ in Amazon ECS.

schreibe /var/lib/docker/tmp/ Blob111111111GetImage: kein Speicherplatz mehr auf dem Gerät

Dieser Fehler weist darauf hin, dass nicht genügend Speicherplatz vorhanden ist.

Um dieses Problem zu lösen, müssen Sie Speicherplatz freigeben.

Wenn Sie das Amazon ECS-optimierte AMI verwenden, können Sie den folgenden Befehl verwenden, um die 20 größten Dateien in Ihrem Dateisystem abzurufen:

```
du -Sh / | sort -rh | head -20
```

Beispielausgabe:

```
5.7G    /var/lib/docker/
containers/50501b5f4cbf90b406e0ca60bf4e6d4ec8f773a6c1d2b451ed8e0195418ad0d2
1.2G    /var/log/ecs
594M    /var/lib/docker/devicemapper/mnt/
c8e3010e36ce4c089bf286a623699f5233097ca126ebd5a700af023a5127633d/rootfs/data/logs
...
```

In einigen Fällen kann das Root-Volume durch einen laufenden Container aufgefüllt werden. Wenn der Container den Standardprotokolltreiber `json-file` ohne eine `max-size`-Beschränkung verwendet, kann die Protokolldatei einen Großteil des Speicherplatzes belegen. Mit dem Befehl `docker ps` können Sie überprüfen, welcher Container den Speicherplatz belegt. Hierzu wird der Verzeichnisname aus der Ausgabe oben der Container-ID zugewiesen. Zum Beispiel:

```
CONTAINER ID   IMAGE                              COMMAND             CREATED
 STATUS              PORTS                              NAMES
50501b5f4cbf   amazon/amazon-ecs-agent:latest   "/agent"            4 days ago
 Up 4 days                                             ecs-agent
```

Bei Verwendung des Protokolltreibers `json-file` erfasst Docker standardmäßig die Standardausgabe (und Standardfehler) aller Container und schreibt diese in Dateien im JSON-Format. Sie können `max-size` als Protokolltreiberoption festlegen, was vermeidet, dass die Protokolldatei zu groß wird. Weitere Informationen finden Sie unter Configure logging drivers in der Docker-Dokumentation.

Nachfolgend finden Sie einen Ausschnitt aus einer Containerdefinition mit dieser Option:

```
{
    "log-driver": "json-file",
    "log-opts": {
        "max-size": "256m"
    }
}
```

Eine Alternative, wenn Ihre Container-Logs zu viel Festplattenspeicher beanspruchen, ist die Verwendung des `awslogs` Log-Treibers. Der `awslogs` Log-Treiber sendet die Logs an CloudWatch, wodurch Speicherplatz frei wird, der andernfalls für Ihre Container-Logs auf der Container-Instance verwendet werden würde. Weitere Informationen finden Sie unter Amazon ECS-Protokolle senden an CloudWatch .

FEHLER: toomanyrequests: Zu viele Anfragen oder Sie haben Ihr Pull-Rate-Limit erreicht.

Dieser Fehler weist darauf hin, dass es eine Docker Hub-Ratenbegrenzung gibt.

Wenn eine der folgenden Fehlermeldungen angezeigt wird, treffen Sie wahrscheinlich die Tarifbeschränkungen des Docker-Hubs:

Weitere Informationen über die Docker-Hub-Ratenlimits finden Sie unter Grundlegendes zur Begrenzung der Docker-Hub-Rate.

Wenn Sie das Docker Hub-Ratenlimit erhöht haben und Ihre Docker-Pulls für Ihre Container-Instances authentifizieren müssen, finden Sie weitere Informationen unter Private Registry-Authentifizierung für Container-Instances.

Fehlerantwort vom Daemon: ***URL*** abrufen: net/http: Anfrage beim Warten auf Verbindung abgebrochen

Dieser Fehler weist darauf hin, dass bei einer Verbindung das Zeitlimit überschritten wurde, da keine Route zum Internet existiert.

Um dieses Problem zu lösen, können Sie:

- Für Aufgabe in öffentlichen Subnetzen geben Sie beim Starten der Aufgabe ENABLED unter Auto-assign public IP (Öffentliche IP automatisch zuweisen) an. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen.
- Für Aufgaben in privaten Subnetzen geben Sie beim Starten der Aufgabe DISABLED (DEAKTIVIERT) für Auto-assign public IP (Öffentliche IP automatisch zuweisen) an und konfigurieren ein NAT-Gateway in Ihrer VPC, um Anforderungen an das Internet weiterzuleiten. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.

ref pull wurde 1 mal wiederholt: konnte nicht kopiert werden:: Fehler beim Öffnen httpReaderSeeker: unerwarteter Statuscode

Dieser Fehler weist darauf hin, dass beim Kopieren eines Bildes ein Fehler aufgetreten ist.

Lesen Sie einen der folgenden Artikel, um dieses Problem zu beheben:

- Informationen zu Fargate-Aufgaben finden Sie unter Wie löse ich den Fehler „cannotpullcontainererror“ für meine Amazon-ECS-Aufgaben auf Fargate.
- Für andere Aufgaben, siehe Wie löse ich den Fehler „cannotpullcontainererror“ für meine Amazon-ECS-Aufgaben.

Pull-Zugriff verweigert

Dieser Fehler weist darauf hin, dass kein Zugriff auf das Bild besteht.

Um dieses Problem zu beheben, müssen Sie möglicherweise Ihren Docker-Client mit Amazon ECR authentifizieren. Weitere Informationen finden Sie unter Private Registry-Authentifizierung im Amazon ECR-Benutzerhandbuch.

Pull-Befehl fehlgeschlagen: Panik: Laufzeitfehler: ungültige Speicheradresse oder Nullzeiger-Dereferenzierung

Dieser Fehler weist darauf hin, dass aufgrund einer ungültigen Speicheradresse oder einer Nullzeiger-Dereferenzierung kein Zugriff auf das Bild möglich ist.

So beheben Sie dieses Problem

- Vergewissern Sie sich, dass Sie über die Sicherheitsgruppenregeln verfügen, um Amazon S3 zu erreichen.
- Wenn Sie Gateway-Endpunkte verwenden, müssen Sie der Routentabelle eine Route hinzufügen, um auf den Endpunkt zuzugreifen.

Fehler beim Abrufen der Image-Konfiguration/Fehler beim Abrufen der Image-Konfiguration

Dieser Fehler weist darauf hin, dass ein Ratenlimit erreicht wurde oder dass ein Netzwerkfehler vorliegt:

Informationen zur Behebung dieses Problems finden Sie unter Wie kann ich den Fehler "CannotPullContainerError" in meiner Amazon ECS EC2 Launch Type Task beheben?

Kontext abgebrochen

Dieser Fehler weist darauf hin, dass der Kontext abgebrochen wurde.

Die häufige Ursache für diesen Fehler liegt darin, dass die von Ihrer Aufgabe verwendete VPC über keine Route zum Abrufen des Container-Images von Amazon ECR verfügt.

Überprüfung, ob Amazon ECS die Aufgabenkonnektivität beendet hat

Es kann vorkommen, dass eine Aufgabe aufgrund eines Problems mit der Netzwerkverbindung beendet wird. Möglicherweise handelt es sich um ein zeitweiliges Problem, das jedoch höchstwahrscheinlich dadurch verursacht wird, dass die Aufgabe keine Verbindung zu einem Endpunkt herstellen kann.

Die Aufgabenkonnektivität wird getestet

Sie können `AWSSupport-TroubleshootECSTaskFailedToStart` Runbook verwenden, um die Task-Konnektivität zu testen. Wenn Sie das Runbook verwenden, benötigen Sie die folgenden Ressourceninformationen:

- Die Aufgaben-ID

  Verwenden Sie die ID der letzten fehlgeschlagenen Aufgabe.
- Der Cluster, in dem sich die Aufgabe befand

Informationen zur Verwendung des Runnbooks finden Sie `AWSSupport-TroubleshootECSTaskFailedToStart`in der Runbook-Referenz zur AWS Systems Manager Automatisierung.

Das Runbook analysiert die Aufgabe. Sie können die Ergebnisse der folgenden Probleme, die den Start einer Aufgabe verhindern können, im Abschnitt Ausgabe anzeigen:

- Netzwerkkonnektivität zur konfigurierten Container-Registry
- VPC-Endpunktkonnektivität
- Konfiguration der Regeln für Sicherheitsgruppen

Behebung von VPC-Endpunktproblemen

Wenn das `AWSSupport-TroubleshootECSTaskFailedToStart` Runbook-Ergebnis auf das VPC-Endpunktproblem hinweist, überprüfen Sie die folgende Konfiguration:

- Die VPC, auf der Sie den Endpunkt erstellen, muss Private DNS verwenden.

- Stellen Sie sicher, dass Sie in derselben VPC wie die Aufgabe einen AWS PrivateLink Endpunkt für den Service haben, zu dem die Aufgabe keine Verbindung herstellen kann. Weitere Informationen finden Sie in einer der folgenden Seiten:

Service	VPC-Endpunktinformationen für den Service	
Amazon ECR	VPC-Endpunkte mit Amazon ECR-Schnittstelle ()AWS PrivateLink	
Systems Manager	Erstellen Sie einen VPC-Endpunkt	
Secrets Manager	Verwenden eines AWS Systems Manager VPC-Endpunkts	
CloudWatch	CloudWatch VPC-Endpunkt	
Amazon S3	AWS PrivateLink für Amazon S3	

- Konfigurieren Sie eine ausgehende Regel für das Task-Subnetz, die HTTPS auf Port 443 DNS-Verkehr (UDP und TCP) zulässt. Weitere Informationen finden Sie unter Regeln zu einer Sicherheitsgruppe hinzufügen im Amazon Elastic Compute Cloud-Benutzerhandbuch.
- Wenn das Subnetz über eine Netzwerk-ACL verfügt, sind die folgenden ACL-Regeln erforderlich:
  - Eine ausgehende Regel, die Datenverkehr zulässt, der Verkehr auf den Ports 1024-65535 zulässt.
  - Eine eingehende Regel, die TCP-Verkehr auf Port 443 zulässt.

  Informationen zur Konfiguration von Regeln finden Sie unter Steuern des Datenverkehrs zu Subnetzen mithilfe von Netzwerk-ACLs im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Behebung von Netzwerkproblemen

Wenn das `AWSSupport-TroubleshootECSTaskFailedToStart` Runbook-Ergebnis auf ein Netzwerkproblem hinweist, überprüfen Sie die folgende Konfiguration:

Aufgaben, die den awsvpc-Netzwerkmodus in einem öffentlichen Subnetz verwenden

Führen Sie die folgende Konfiguration auf der Grundlage des Runbooks durch:

- Für Aufgabe in öffentlichen Subnetzen geben Sie beim Starten der Aufgabe ENABLED unter Auto-assign public IP (Öffentliche IP automatisch zuweisen) an. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen.
- Sie benötigen ein Gateway, um den Internetverkehr abzuwickeln. Die Routentabelle für das Task-Subnetz muss eine Route für den Verkehr zum Gateway enthalten.

  Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Routen aus einer Routentabelle im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Typ des Gateways	Ziel der Routentabelle	Ziel der Routentabelle
NAT	0.0.0.0/0	NAT-Gateway-ID
Internet-Gateway	0.0.0.0/0	Internet-Gateway-ID

- Wenn das Task-Subnetz über eine Netzwerk-ACL verfügt, sind die folgenden ACL-Regeln erforderlich:
  - Eine ausgehende Regel, die Datenverkehr zulässt, der Verkehr auf den Ports 1024-65535 zulässt.
  - Eine eingehende Regel, die TCP-Verkehr auf Port 443 zulässt.

  Informationen zur Konfiguration von Regeln finden Sie unter Steuern des Datenverkehrs zu Subnetzen mithilfe von Netzwerk-ACLs im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Aufgaben, die den awsvpc-Netzwerkmodus in einem privaten Subnetz verwenden

Führen Sie die folgende Konfiguration auf der Grundlage des Runbooks durch:

- Wählen Sie DISABLED für Öffentliche IP automatisch zuweisen, wenn Sie die Aufgabe starten.
- Konfigurieren Sie ein NAT-Gateway in Ihrer VPC, um Anfragen an das Internet weiterzuleiten. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.
- Die Routentabelle für das Task-Subnetz muss eine Route für den Verkehr zum NAT-Gateway enthalten.

Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Routen aus einer Routentabelle im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Typ des Gateways	Ziel der Routentabelle	Ziel der Routentabelle
NAT	0.0.0.0/0	NAT-Gateway-ID

- Wenn das Task-Subnetz über eine Netzwerk-ACL verfügt, sind die folgenden ACL-Regeln erforderlich:
  - Eine ausgehende Regel, die Datenverkehr zulässt, der Verkehr auf den Ports 1024-65535 zulässt.
  - Eine eingehende Regel, die TCP-Verkehr auf Port 443 zulässt.

  Informationen zur Konfiguration von Regeln finden Sie unter Steuern des Datenverkehrs zu Subnetzen mithilfe von Netzwerk-ACLs im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Aufgaben, die den awsvpc-Netzwerkmodus in einem öffentlichen Subnetz nicht verwenden

Führen Sie die folgende Konfiguration auf der Grundlage des Runbooks durch:

- Wählen Sie bei der Erstellung des Clusters unter Netzwerk für Amazon EC2 EC2-Instances die Option Einschalten für automatische IP-Zuweisung.

  Diese Option weist der primären Netzwerkschnittstelle der Instance eine öffentliche IP-Adresse zu.
- Sie benötigen ein Gateway, um den Internetverkehr abzuwickeln. Die Routentabelle für das Instanz-Subnetz muss eine Route für den Verkehr zum Gateway enthalten.

  Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Routen aus einer Routentabelle im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Typ des Gateways	Ziel der Routentabelle	Ziel der Routentabelle
NAT	0.0.0.0/0	NAT-Gateway-ID
Internet-Gateway	0.0.0.0/0	Internet-Gateway-ID

- Wenn das Instance-Subnetz über eine Netzwerk-ACL verfügt, sind die folgenden ACL-Regeln erforderlich:

- Eine ausgehende Regel, die Datenverkehr zulässt, der Verkehr auf den Ports 1024-65535 zulässt.
- Eine eingehende Regel, die TCP-Verkehr auf Port 443 zulässt.

Informationen zur Konfiguration von Regeln finden Sie unter Steuern des Datenverkehrs zu Subnetzen mithilfe von Netzwerk-ACLs im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Aufgaben, die den awsvpc-Netzwerkmodus in einem privaten Subnetz verwenden

Führen Sie die folgende Konfiguration auf der Grundlage des Runbooks durch:

- Wählen Sie bei der Erstellung des Clusters unter Netzwerk für Amazon EC2 EC2-Instances die Option Ausschalten für Automatische IP-Zuweisung.
- Konfigurieren Sie ein NAT-Gateway in Ihrer VPC, um Anfragen an das Internet weiterzuleiten. Weitere Informationen finden Sie unter NAT-Gateways im Amazon VPC-Benutzerhandbuch.
- Die Routentabelle für das Instanz-Subnetz muss eine Route für den Verkehr zum NAT-Gateway enthalten.

  Weitere Informationen finden Sie unter Hinzufügen und Entfernen von Routen aus einer Routentabelle im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

Typ des Gateways	Ziel der Routentabelle	Ziel der Routentabelle
NAT	0.0.0.0/0	NAT-Gateway-ID

- Wenn das Task-Subnetz über eine Netzwerk-ACL verfügt, sind die folgenden ACL-Regeln erforderlich:
  - Eine ausgehende Regel, die Datenverkehr zulässt, der Verkehr auf den Ports 1024-65535 zulässt.
  - Eine eingehende Regel, die TCP-Verkehr auf Port 443 zulässt.

  Informationen zur Konfiguration von Regeln finden Sie unter Steuern des Datenverkehrs zu Subnetzen mithilfe von Netzwerk-ACLs im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

IAM-Rollenanfragen für Amazon ECS-Aufgaben anzeigen

Wenn Sie in einer IAM-Rolle einen Anbieter für Ihre Aufgabenanmeldedaten verwenden, werden die Anbieteranfragen in einem Auditprotokoll gespeichert. Das Überwachungsprotokoll übernimmt dieselben Protokollrotationseinstellungen wie das Container-Agent-Protokoll. Die Konfigurationsvariablen `ECS_LOG_ROLLOVER_TYPE`, `ECS_LOG_MAX_FILE_SIZE_MB` und `ECS_LOG_MAX_ROLL_COUNT` des Container-Agenten können so eingestellt werden, dass sie das Verhalten des Überwachungsprotokolls beeinflussen. Weitere Informationen finden Sie unter Konfigurationsparameter für das Amazon ECS-Container-Agent-Protokoll.

Für Container-Agent Version 1.36.0 und höher befindet sich das Überwachungsprotokoll unter `/var/log/ecs/audit.log`. Wenn das Protokoll rotiert wird, wird am Ende des Protokolldateinamens ein Zeitstempel im *YYYY*-*MM*-*DD*-*HH*-Format hinzugefügt.

Für Container-Agent Version 1.35.0 und früher befindet sich das Überwachungsprotokoll unter `/var/log/ecs/audit.log.`*YYYY*-*MM*-*DD*-*HH*.

Das Format des Protokolleintrags ist wie folgt:

- Zeitstempel
- HTTP-Antwortcode
- IP-Adresse und Portnummer von Abfrageursprung
- Relative URI des Anmeldeinformationsanbieters
- Der Benutzeragent, der die Abfrage gesendet hat
- Der ARN der Aufgabe, zu dem der abfragende Container gehört
- Der `GetCredentials`-API-Name und Versionsnummer
- Der Name des Amazon-ECS-Clusters, für den die Container-Instance registriert ist
- Der ARN der Container-Instance

Sie können den folgenden Befehl verwenden, um die Protokolldateien anzuzeigen.

```
cat /var/log/ecs/audit.log.2016-07-13-16
```

Ausgabe:

```
2016-07-13T16:11:53Z 200 172.17.0.5:52444 "/v1/credentials" "python-requests/2.7.0
 CPython/2.7.6 Linux/4.4.14-24.50.amzn1.x86_64" TASK_ARN GetCredentials
 1 CLUSTER_NAME CONTAINER_INSTANCE_ARN
```

Ereignismeldungen des Amazon ECS-Service anzeigen

Wenn Sie ein Problem mit einem Service beheben möchten, sollten Sie zuerst im Service-Ereignisprotokoll nach Diagnoseinformationen suchen. Sie können Serviceereignisse mithilfe der `DescribeServices` API AWS CLI, der oder mithilfe von anzeigen AWS Management Console.

Beim Anzeigen von Service-Ereignismeldungen mit der Amazon-ECS-API werden nur die Ereignisse aus dem Service-Scheduler zurückgegeben. Dazu gehören die aktuellste Aufgabenplatzierung und Instance-Integritätsereignisse. Die Amazon-ECS-Konsole zeigt jedoch Service-Ereignisse aus den folgenden Quellen an.

- Aufgabenplatzierung und Instance-Integritätsereignisse aus dem Amazon-ECS-Service-Scheduler. Diese Ereignisse haben das Präfix service ***(service-name)***. Um sicherzustellen, dass diese Ereignisansicht hilfreich ist, zeigen wir nur die `100` aktuellsten Ereignisse an und doppelte Ereignismeldungen werden weggelassen, bis entweder die Ursache behoben ist oder sechs Stunden vergangen sind. Wenn die Ursache nicht innerhalb von sechs Stunden behoben ist, erhalten Sie eine weitere Serviceereignismeldung für diese Ursache.
- Service Auto Scaling-Events. Diese Ereignisse haben das Präfix Message. Die `10` letzten Skalierungsereignisse werden angezeigt. Diese Ereignisse treten nur auf, wenn ein Service mit einer Application Auto Scaling-Skalierungsrichtlinie konfiguriert ist.

Gehen Sie wie folgt vor, um Ihre aktuellen Service-Ereignismeldungen anzuzeigen.

Console

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Klicken Sie im Navigationsbereich auf Cluster.
3. Wählen Sie auf der Cluster-Seite den Cluster aus.
4. Wählen Sie den zu untersuchenden Service aus.
5. Wählen Sie Deployments and events (Bereitstellungen und Ereignisse) und zeigen Sie unter Events (Ereignisse) die Meldungen an.

AWS CLI

Verwenden Sie den Befehl describe-services, um die Serviceereignismeldungen für einen angegebenen Service anzuzeigen.

Im folgenden AWS CLI Beispiel wird der Service Service *Name* im *Standardcluster* beschrieben, der die neuesten Service-Event-Meldungen bereitstellt.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Amazon ECS-Serviceereignismeldungen

Im Folgenden finden Sie Beispiele für Service-Ereignismeldungen, die in der Amazon-ECS-Konsole angezeigt werden können:

service (***service-name***) hat einen stabilen Zustand erreicht.

Der Dienstplaner sendet ein `service (service-name) has reached a steady state.` Serviceereignis, wenn der Dienst fehlerfrei ist und die gewünschte Anzahl von Aufgaben ausgeführt wurde, wodurch ein stabiler Zustand erreicht wird.

Der Service-Scheduler meldet den Status regelmäßig, sodass Sie diese Nachricht möglicherweise mehrmals erhalten.

Service (***Servicename***) konnte eine Aufgabe nicht platzieren, da keine Container-Instance alle Anforderungen erfüllt hat.

Der Service Scheduler sendet diese Ereignisnachricht, wenn er die verfügbaren Ressourcen zum Hinzufügen einer weiteren Aufgabe nicht finden konnte. Mögliche Ursachen dafür sind:

Es wurden keine Container-Instances in Ihrem Cluster gefunden

Wenn in dem Cluster, in dem Sie versuchen, eine Aufgabe auszuführen, keine Container-Instances registriert sind, erhalten Sie diesen Fehler. Sie sollten Ihrem Cluster Container-Instances hinzufügen. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

Nicht genügend Ports

Wenn Ihre Aufgabe feste Host-Port-Zuweisung verwendet (wenn zum Beispiel Ihre Aufgabe Port 80 auf dem Host für einen Webserver verwendet), brauchen Sie mindestens eine Container-Instance pro Aufgabe, weil nur ein Container einen einzelnen Host-Port auf einmal verwenden kann. Sie sollten Ihrem Cluster Container-Instances hinzufügen oder die Anzahl gewünschter Aufgaben reduzieren.

Registrierung einer zu großen Zahl von Ports

Die Container-Instance, die der Aufgabenplatzierung am nächsten kommt, darf die maximal zulässige Obergrenze für reservierte Ports von 100 Host-Ports pro Container-Instance nicht überschreiten. Ein dynamisches Host-Port-Mapping behebt das Problem möglicherweise.

Port wird bereits verwendet

Die Aufgabendefinition dieser Aufgabe verwendet in ihrer Portzuordnung denselben Port wie eine Aufgabe, die bereits auf der ausgewählten Container-Instance ausgeführt wird. Die Serviceereignis-Nachricht hätte die gewählte Container-Instance-ID als Teil der folgenden Nachricht.

```
The closest matching container-instance is already using a port required by your
 task.
```

Speicher reicht nicht aus

Wenn Ihre Aufgabendefinition 1000 MiB Speicher angibt und jede Container-Instance in Ihrem Cluster 1024 MiB Speicher hat, können Sie nur eine Kopie dieser Aufgabe pro Container-Instance ausführen. Sie können mit weniger Speicher in Ihrer Aufgabendefinition experimentieren, sodass Sie mehr als eine Aufgabe pro Container-Instance oder mehr Container-Instances in Ihrem Cluster starten können.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Aufgaben so viel Arbeitsspeicher wie möglich für einen bestimmten Instance-Typ zuweisen, lesen Sie nach unter Reservieren von Amazon ECS Linux-Container-Instance-Speicher .

CPU reicht nicht aus

Eine Container-Instance hat 1 024 CPU-Einheiten für jeden CPU-Kern. Wenn Ihre Aufgabendefinition 1.000 CPU-Einheiten angibt und jede Container-Instance in Ihrem Cluster 1 024 CPU-Einheiten hat, können Sie nur eine Kopie dieser Aufgabe pro Container-Instance ausführen. Sie können mit weniger CPU-Einheiten in Ihrer Aufgabendefinition experimentieren, sodass Sie mehr als eine Aufgabe pro Container-Instance oder mehr Container-Instances in Ihrem Cluster starten können.

Nicht genügend verfügbare ENI-Befestigungspunkte

Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, erhalten ihre eigene Elastic-Network-Schnittstelle (ENI), die an die Container-Instance angehängt wird, die sie hostet. Amazon-EC2-Instances haben eine Begrenzung für die Anzahl der ENIs, die an sie angehängt werden können, und es gibt keine Container-Instances mit verfügbarer ENI-Kapazität im Cluster.

Das ENI-Limit für einzelne Container-Instances hängt von den folgenden Bedingungen ab:

- Wenn Sie sich nicht für die `awsvpcTrunking`-Kontoeinstellung angemeldet haben, hängt das ENI-Limit für jede Container-Instance vom Instance-Typ ab. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2-Benutzerhandbuch.
- Wenn Sie sich für die `awsvpcTrunking` Kontoeinstellungen entschieden haben, aber nach der Anmeldung keine neuen Container-Instances mit einem unterstützten Instance-Typ gestartet haben, ist das ENI-Limit für jede Container-Instance immer noch auf dem Standardwert. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2-Benutzerhandbuch.
- Wenn Sie sich für die `awsvpcTrunking`-Kontoeinstellung angemeldet haben und Sie haben neue Container-Instances mit einem unterstützten Instance-Typ nach der Anmeldung gestartet, stehen zusätzliche ENIs zur Verfügung. Weitere Informationen finden Sie unter Unterstützte Instances für mehr Amazon ECS-Container-Netzwerkschnittstellen.

Weitere Informationen zum Aktivieren der `awsvpcTrunking`-Kontoeinstellung finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.

Sie können Ihrem Cluster Container-Instances hinzufügen, um weitere Netzwerkadapter zur Verfügung zu stellen.

Container-Instance fehlt erforderliches Attribut

Einige Aufgabendefinitionsparameter erfordern, dass eine bestimmte Docker-Remote-API-Version auf der Container-Instance installiert wird. Andere, wie die Optionen für Protokolltreiber, erfordern, dass die Container-Instances diese Protokolltreiber bei der Variablen zur Konfiguration des Agenten `ECS_AVAILABLE_LOGGING_DRIVERS` registrieren. Wenn Ihre Aufgabendefinition einen Parameter enthält, der ein bestimmtes Container-Instance-Attribut erfordert, und Sie keine verfügbaren Container-Instances haben, die diese Anforderung erfüllen können, kann die Aufgabe nicht platziert werden.

Eine häufige Ursache für diesen Fehler ist, dass Ihr Service Tasks verwendet, die den `awsvpc` Netzwerkmodus und den EC2-Starttyp verwenden. Für den von Ihnen angegebenen Cluster ist keine Container-Instance in demselben Subnetz registriert, das `awsvpcConfiguration` bei der Erstellung des Dienstes angegeben wurde.

Weitere Informationen darüber, welche Attribute für bestimmte Aufgabendefinitionsparameter und Agentenkonfigurationsvariablen erforderlich sind, finden Sie unter Amazon ECS-Aufgabendefinitionsparameter und Konfiguration des Amazon-ECS-Container-Agenten.

Service (***`Servicename`***) konnte eine Aufgabe nicht platzieren, da keine Container-Instance alle Anforderungen erfüllt hat. Die am besten passende Container-Instance ***`container-instance-id`*** verfügt nicht über ausreichend CPU-Einheiten.

Die Container-Instance, die der Aufgabenplatzierung am ehesten entspricht, enthält nicht genügend CPU-Einheiten, um die Anforderungen in der Aufgabendefinition zu erfüllen. Überprüfen Sie die CPU-Anforderungen sowohl in den Aufgabengrößen- als auch den Containerdefinitionsparametern der Aufgabendefinition.

Service (***`Servicename`***) konnte eine Aufgabe nicht platzieren, da keine Container-Instance alle Anforderungen erfüllt hat. Für die am besten passende Container-Instance ***`Container-Instance-ID`*** ist der Fehler "AGENT" aufgetreten.

Der Amazon-ECS-Container-Agent auf der am besten passenden Container-Instance für die Aufgabenplatzierung wird getrennt. Wenn Sie mit SSH eine Verbindung zu der Container-Instance herstellen können, können Sie die Agenten-Protokolle überprüfen. Weitere Informationen finden Sie unter Konfigurationsparameter für das Amazon ECS-Container-Agent-Protokoll. Sie sollten auch prüfen, ob der Agent auf der Instance ausgeführt wird. Wenn Sie das Amazon-ECS-optimierte AMI

verwenden, können Sie versuchen, den Agenten mit dem folgenden Befehl zu stoppen und neu zu starten.

- Für das Amazon ECS-optimierte Amazon Linux 2 AMI und das Amazon ECS-optimierte Amazon Linux 2023 AMI

```
sudo systemctl restart ecs
```

- Für das Amazon-ECS-optimierte Amazon-Linux-AMI

```
sudo stop ecs && sudo start ecs
```

Service (***Service-Name***) (***Instance-ID***) ist in (elb ***elb-name***) fehlerhaft. Grund: Die Instance hat mindestens die Anzahl der Integritätsprüfungen nacheinander nicht bestanden. UnhealthyThreshold

Dieser Service ist mit einem Load Balancer registriert und die Zustandsprüfungen des Load Balancer sind fehlgeschlagen. Weitere Informationen finden Sie unter Fehlerbehebung bei Service Load Balancers in Amazon ECS.

Service (***service-name***) kann nicht alle Aufgaben erfolgreich starten.

Dieser Service enthält Aufgaben, die nach mehrmaligen Versuchen nicht gestartet werden konnten. Zu diesem Zeitpunkt beginnt der Service-Scheduler, die Zeit zwischen erneuten Versuchen inkrementell zu erhöhen. Sie sollten eine Fehlersuche durchführen, um festzustellen, warum Ihre Aufgaben nicht starten. Weitere Informationen finden Sie unter Drosselungslogik für Amazon ECS-Services.

Nachdem der Service aktualisiert wurde, z. B. durch eine aktualisierte Aufgabendefinition, nimmt der Service-Scheduler sein normales Verhalten wieder auf.

Service (***service-name***) Operationen werden gedrosselt. Wird es später neu versuchen.

Dieser Service kann aufgrund von API-Beschränkungen keine weiteren Aufgaben starten. Sobald der Service-Scheduler in der Lage ist, weitere Aufgaben zu starten, wird er fortgesetzt.

Wenn Sie eine Kontingenterhöhung des API-Ratenlimits beantragen möchten, öffnen Sie die Seite AWS Support -Center, melden sich gegebenenfalls an und wählen Create case (Fall erstellen).

Wählen Sie Service Limit increase (Erhöhung des Servicelimits). Füllen Sie das Formular aus und senden Sie es ab.

Service (***service-name***) konnte Tasks während einer Bereitstellung aufgrund der Konfiguration der Dienstbereitstellung nicht anhalten oder starten. Aktualisieren Sie den Wert minimumHealthyPercent oder MaximumPercent und versuchen Sie es erneut.

Dieser Service kann Aufgaben während einer Servicebereitstellung aufgrund der Bereitstellungskonfiguration nicht anhalten oder starten. Die Bereitstellungskonfiguration besteht aus den `maximumPercent` Werten `minimumHealthyPercent` und, die bei der Erstellung des Dienstes definiert werden. Diese Werte können auch für einen vorhandenen Dienst aktualisiert werden.

Das `minimumHealthyPercent` stellt die Untergrenze für die Anzahl der Aufgaben dar, die für einen Dienst während einer Bereitstellung oder wenn eine Container-Instance leer ist, ausgeführt werden sollten. Das ist ein Prozent der gewünschten Anzahl von Aufgaben für den Service. Dieser Wert wird aufgerundet. Wenn beispielsweise die Mindestanzahl fehlerfreier Aufgaben bei vier liegt `50` und die gewünschte Anzahl an Aufgaben bei vier liegt, kann der Planer zwei bestehende Aufgaben beenden, bevor zwei neue Aufgaben gestartet werden. Ebenso kann der Scheduler, wenn der minimale fehlerfreie Prozentsatz 75 % beträgt und die gewünschte Anzahl zwei ist, keine Aufgaben stoppen, da der resultierende Wert auch zwei ist.

Der `maximumPercent` stellt die Obergrenze für die Anzahl der Aufgaben dar, die für einen Dienst während einer Bereitstellung oder wenn eine Container-Instance ausgelastet ist, ausgeführt werden sollten. Das ist ein Prozent der gewünschten Anzahl von Aufgaben für einen Service. Dieser Wert wird abgerundet. Wenn der maximale Prozentsatz beispielsweise vier beträgt `200` und die gewünschte Anzahl der Aufgaben bei vier liegt, kann der Scheduler vier neue Aufgaben starten, bevor er vier bestehende Aufgaben stoppt. Ebenso, wenn der maximale Prozentsatz `125`. ist und die gewünschte Aufgabenanzahl drei ist, kann der Scheduler keine Aufgaben starten, da der resultierende Wert ebenfalls drei ist.

Wenn Sie einen minimalen fehlerfreien Prozentsatz oder einen maximalen Prozentsatz festlegen, sollten Sie sicherstellen, dass der Scheduler mindestens eine Aufgabe anhalten oder starten kann, wenn eine Bereitstellung ausgelöst wird.

Service (*service-name*) konnte keine Aufgabe platzieren. Grund: Sie haben das Limit für die Anzahl der Aufgaben erreicht, die Sie gleichzeitig ausführen können

Sie können eine Kontingenterhöhung für die Ressource anfordern, die den Fehler verursacht hat. Weitere Informationen finden Sie unter Servicekontingente. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter Anfordern einer Kontingenterhöhung im Benutzerhandbuch zu Service Quotas.

Service (*service-name*) konnte keine Aufgabe platzieren. Grund: Interner Fehler.

Im Folgenden finden Sie die möglichen Gründe für diesen Fehler:

- Der Service kann eine Aufgabe nicht starten, da sich ein Subnetz in einer nicht unterstützten Availability Zone befindet.

  Informationen zu den unterstützten Fargate-Regionen und Availability Zones finden Sie unter the section called “AWS Fargate-Regionen”.

  Weitere Informationen zum Anzeigen der Subnetz-Availability-Zone finden Sie unter Anzeigen Ihres Subnetzes im Benutzerhandbuch zu Amazon VPC.
- Sie versuchen, eine Aufgabendefinition auszuführen, die die ARM-Architektur auf Fargate Spot verwendet.

Service (*service-name*) konnte keine Aufgabe platzieren. Grund: Die angeforderte CPU-Konfiguration liegt über Ihrem Limit.

Sie können eine Kontingenterhöhung für die Ressource anfordern, die den Fehler verursacht hat. Weitere Informationen finden Sie unter Servicekontingente. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter Anfordern einer Kontingenterhöhung im Benutzerhandbuch zu Service Quotas.

Service (*service-name*) konnte keine Aufgabe platzieren. Grund: Die angeforderte MEMORY-Konfiguration liegt über Ihrem Limit.

Sie können eine Kontingenterhöhung für die Ressource anfordern, die den Fehler verursacht hat. Weitere Informationen finden Sie unter Servicekontingente. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter Anfordern einer Kontingenterhöhung im Benutzerhandbuch zu Service Quotas.

Service (*service-name*) konnte keine Aufgabe platzieren. Grund: Sie haben das Limit für die Anzahl der vCPUs erreicht, die Sie gleichzeitig ausführen können

AWS Fargate geht von Quoten auf Basis der Task-Anzahl auf vCPU-basierte Kontingente über.

Sie können eine Kontingenterhöhung für das vCPU-basierte Kontingent von Fargate anfordern. Weitere Informationen finden Sie unter Servicekontingente. Informationen zur Erhöhung eines Fargate-Kontingents finden Sie unter Anfordern einer Kontingenterhöhung im Service-Quotas-Benutzerhandbuch.

Der Service (*service-name*) konnte den stabilen Status nicht erreichen, da der Aufgabensatz (*taskSet-ID*) nicht abskaliert werden konnte. Grund: Die Anzahl der geschützten Aufgaben übersteigt die gewünschte Anzahl von Aufgaben.

Der Service verfügt über mehr geschützte Aufgaben als die gewünschte Anzahl von Aufgaben. Sie können eine der folgenden Aktionen durchführen:

- Warten Sie, bis der Schutz für die aktuellen Aufgaben abgelaufen ist, damit diese beendet werden können.
- Ermitteln Sie, welche Aufgaben gestoppt werden können, und verwenden Sie die `UpdateTaskProtection` API mit der `protectionEnabled` Option, `false` um den Schutz für diese Aufgaben aufzuheben.
- Erhöhen Sie die Anzahl der gewünschten Aufgaben des Services auf mehr als die Anzahl der geschützten Aufgaben.

Service (*service-name*) konnte keinen stabilen Zustand erreichen. Grund: In Ihrem Kapazitätsanbieter wurden keine Container-Instances gefunden.

Der Service Scheduler sendet diese Ereignisnachricht, wenn er die verfügbaren Ressourcen zum Hinzufügen einer weiteren Aufgabe nicht finden konnte. Mögliche Ursachen dafür sind:

Dem Cluster ist kein Kapazitätsanbieter zugeordnet

> Verwenden Sie diese Option, `describe-services` um zu überprüfen, ob dem Cluster ein Kapazitätsanbieter zugeordnet ist. Sie können die Kapazitätsanbieterstrategie für den Service aktualisieren.

Stellen Sie sicher, dass im Kapazitätsanbieter Kapazität verfügbar ist. Stellen Sie beim EC2-Starttyp sicher, dass die Container-Instances die Anforderungen der Aufgabendefinition erfüllen.

Es wurden keine Container-Instances in Ihrem Cluster gefunden

Wenn in dem Cluster, in dem Sie versuchen, eine Aufgabe auszuführen, keine Container-Instances registriert sind, erhalten Sie diesen Fehler. Sie sollten Ihrem Cluster Container-Instances hinzufügen. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.

Nicht genügend Ports

Wenn Ihre Aufgabe eine feste Host-Port-Zuordnung verwendet (Ihre Aufgabe verwendet beispielsweise Port 80 auf dem Host für einen Webserver), benötigen Sie mindestens eine Container-Instance pro Aufgabe. Nur ein Container kann jeweils einen einzelnen Host-Port verwenden. Sie sollten Ihrem Cluster Container-Instances hinzufügen oder die Anzahl gewünschter Aufgaben reduzieren.

Registrierung einer zu großen Zahl von Ports

Die Container-Instance, die der Aufgabenplatzierung am nächsten kommt, darf die maximal zulässige Obergrenze für reservierte Ports von 100 Host-Ports pro Container-Instance nicht überschreiten. Ein dynamisches Host-Port-Mapping behebt das Problem möglicherweise.

Port wird bereits verwendet

Die Aufgabendefinition dieser Aufgabe verwendet in ihrer Portzuordnung denselben Port wie eine Aufgabe, die bereits auf der ausgewählten Container-Instance ausgeführt wird. Die Serviceereignis-Nachricht hätte die gewählte Container-Instance-ID als Teil der folgenden Nachricht.

```
The closest matching container-instance is already using a port required by your
 task.
```

Speicher reicht nicht aus

Wenn Ihre Aufgabendefinition 1000 MiB Speicher angibt und jede Container-Instance in Ihrem Cluster 1024 MiB Speicher hat, können Sie nur eine Kopie dieser Aufgabe pro Container-Instance ausführen. Sie können mit weniger Speicher in Ihrer Aufgabendefinition experimentieren, sodass Sie mehr als eine Aufgabe pro Container-Instance oder mehr Container-Instances in Ihrem Cluster starten können.

> **Note**
>
> Wenn Sie versuchen, Ihre Ressourcennutzung zu maximieren, indem Sie Ihren Aufgaben so viel Arbeitsspeicher wie möglich für einen bestimmten Instance-Typ zuweisen, lesen Sie nach unter Reservieren von Amazon ECS Linux-Container-Instance-Speicher .

Nicht genügend verfügbare ENI-Befestigungspunkte

Aufgaben, die den Netzwerkmodus `awsvpc` verwenden, erhalten ihre eigene Elastic-Network-Schnittstelle (ENI), die an die Container-Instance angehängt wird, die sie hostet. Amazon EC2 EC2-Instances haben ein Limit für die Anzahl der ENIs, die an sie angehängt werden können, und es gibt keine Container-Instances im Cluster, für die ENI-Kapazität verfügbar ist.

Das ENI-Limit für einzelne Container-Instances hängt von den folgenden Bedingungen ab:

- Wenn Sie sich nicht für die `awsvpcTrunking`-Kontoeinstellung angemeldet haben, hängt das ENI-Limit für jede Container-Instance vom Instance-Typ ab. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2-Benutzerhandbuch.
- Wenn Sie sich für die `awsvpcTrunking` Kontoeinstellungen entschieden haben, aber nach der Anmeldung keine neuen Container-Instances mit einem unterstützten Instance-Typ gestartet haben, ist das ENI-Limit für jede Container-Instance immer noch auf dem Standardwert. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2-Benutzerhandbuch.
- Wenn Sie sich für die `awsvpcTrunking`-Kontoeinstellung angemeldet haben und Sie haben neue Container-Instances mit einem unterstützten Instance-Typ nach der Anmeldung gestartet, stehen zusätzliche ENIs zur Verfügung. Weitere Informationen finden Sie unter Unterstützte Instances für mehr Amazon ECS-Container-Netzwerkschnittstellen.

Weitere Informationen zum Aktivieren der `awsvpcTrunking`-Kontoeinstellung finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.

Sie können Ihrem Cluster Container-Instances hinzufügen, um weitere Netzwerkadapter zur Verfügung zu stellen.

Container-Instance fehlt erforderliches Attribut

Einige Aufgabendefinitionsparameter erfordern, dass eine bestimmte Docker-Remote-API-Version auf der Container-Instance installiert wird. Andere, wie die Optionen für Protokolltreiber,

erfordern, dass die Container-Instances diese Protokolltreiber bei der Variablen zur Konfiguration des Agenten `ECS_AVAILABLE_LOGGING_DRIVERS` registrieren. Wenn Ihre Aufgabendefinition einen Parameter enthält, der ein bestimmtes Container-Instance-Attribut erfordert, und Sie keine verfügbaren Container-Instances haben, die diese Anforderung erfüllen können, kann die Aufgabe nicht platziert werden.

Eine häufige Ursache für diesen Fehler ist, wenn Ihr Service Tasks verwendet, die den `awsvpc` Netzwerkmodus und den EC2-Starttyp verwenden, und für den von Ihnen angegebenen Cluster keine Container-Instance in demselben Subnetz registriert ist, das `awsvpcConfiguration` bei der Erstellung des Dienstes angegeben wurde.

Weitere Informationen darüber, welche Attribute für bestimmte Aufgabendefinitionsparameter und Agentenkonfigurationsvariablen erforderlich sind, finden Sie unter Amazon ECS-Aufgabendefinitionsparameter und Konfiguration des Amazon-ECS-Container-Agenten.

Service (***`service-name`***) konnte keine Aufgabe platzieren. Grund: Kapazität ist derzeit nicht verfügbar. Bitte versuchen Sie es später erneut oder in einer anderen Availability Zone.

Derzeit ist keine Kapazität verfügbar, mit der Ihr Service ausgeführt werden kann.

Sie können eine der folgenden Aktionen durchführen:

- Warten Sie, bis die Fargate-Kapazität- oder EC2-Container-Instances verfügbar sind.
- Starten Sie den Service neu und geben Sie zusätzliche Subnetze an.

Die Bereitstellung des Dienstes (***`Dienstname`***) ist fehlgeschlagen: Aufgaben konnten nicht gestartet werden.

Die Aufgaben in Ihrem Dienst konnten nicht gestartet werden.

Informationen zum Debuggen gestoppter Aufgaben finden Sie unter. Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS

service (*service-name) Beim* Warten auf den Start von Amazon ECS Agent ist eine Zeitüberschreitung aufgetreten. Bitte überprüfen Sie die Protokolle unter /var/log/ecs/ecs-agent.log“.

Der Amazon-ECS-Container-Agent auf der am besten passenden Container-Instance für die Aufgabenplatzierung wird getrennt. Wenn Sie mit SSH eine Verbindung zur Container-Instance herstellen können, können Sie die Agentenprotokolle überprüfen. Weitere Informationen finden Sie unter Konfigurationsparameter für das Amazon ECS-Container-Agent-Protokoll. Sie sollten auch prüfen, ob der Agent auf der Instance ausgeführt wird. Wenn Sie das Amazon-ECS-optimierte AMI verwenden, können Sie versuchen, den Agenten mit dem folgenden Befehl zu stoppen und neu zu starten.

- Für das Amazon-ECS-optimierte Amazon-Linux-2-AMI

```
sudo systemctl restart ecs
```

- Für das Amazon-ECS-optimierte Amazon-Linux-AMI

```
sudo stop ecs && sudo start ecs
```

*Der Tasksatz von service (service-name) (TaskSet-ID) ist in der Zielgruppe (Targetgroup-ARN) nicht fehlerfrei aufgrund von.*TARGET GROUP IS NOT FOUND

Die für den Service festgelegte Aufgabe hat die Zustandsprüfungen nicht bestanden, weil die Zielgruppe nicht gefunden wurde. Sie sollten den Dienst löschen und neu erstellen. Löschen Sie keine Elastic Load Balancing Balancing-Zielgruppe, es sei denn, der entsprechende Amazon ECS-Service wurde bereits gelöscht.

*Der Tasksatz von service (service-name) (TaskSet-ID) ist in der Zielgruppe (Targetgroup-ARN) nicht fehlerfrei aufgrund von.*TARGET IS NOT FOUND

Die für den Service festgelegte Aufgabe hat die Integritätsprüfungen nicht bestanden, weil das Ziel nicht gefunden wurde.

Fehlerbehebung bei Service Load Balancers in Amazon ECS

Amazon-ECS-Services können Aufgaben bei einem Elastic Load Balancing-Load Balancer registrieren. Fehler bei der Konfiguration von Load Balancers sind häufig die Ursache für gestoppte Aufgaben. Wenn Ihre gestoppten Aufgaben von Services gestartet wurden, die einen Load Balancer verwenden, ziehen Sie folgende mögliche Ursachen in Betracht.

Die mit dem Service verknüpfte Amazon ECS-Rolle ist nicht vorhanden

Die Amazon ECS servicegebundene Rolle ermöglicht es Amazon-ECS-Services Container-Instances mit Elastic Load Balancing-Load Balancern zu registrieren. Die servicegebundene Rolle muss in Ihrem Konto erstellt werden. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.

Sicherheitsgruppe für Container-Instances

Wenn Ihr Container dem Port 80 auf Ihrer Container-Instance zugewiesen ist, muss Ihre Container-Instance-Sicherheitsgruppe eingehenden Datenverkehr auf Port 80 für die Zustandsprüfungen des Load Balancer erlauben.

Elastic Load Balancing Load Balancer ist nicht für alle Availability Zones konfiguriert

Ihr Load Balancer sollte so konfiguriert sein, dass er alle Availability Zones in einer Region verwenden kann, oder zumindest alle Availability Zones, in denen sich Ihre Container-Instances befinden. Wenn ein Service einen Load Balancer verwendet und eine Aufgabe auf einer Container-Instance startet, die sich in einer Availability Zone befindet, für deren Verwendung der Load Balancer nicht konfiguriert ist, besteht die Aufgabe die Zustandsprüfung nie. Dies führt dazu, dass die Aufgabe beendet wird.

Elastic Load Balancing Load Balancer Health Check falsch konfiguriert

Die Parameter der Zustandsprüfung des Load Balancer können übermäßig restriktiv sein oder auf Ressourcen zeigen, die nicht existieren. Wenn festgestellt wird, dass eine Container-Instance fehlerhaft ist, wird sie aus dem Load Balancer entfernt. Stellen Sie sicher, dass die folgenden Parameter korrekt für Ihren Service-Load Balancer konfiguriert sind.

Ping-Port

Der Wert Ping-Port für eine Load Balancer-Zustandsprüfung ist der Port auf den Container-Instances, den der Load Balancer prüft, um festzustellen, ob er fehlerfrei ist. Wenn dieser Port falsch konfiguriert ist, meldet der Load Balancer Ihre Container-Instance wahrscheinlich von sich ab. Dieser Port sollte so konfiguriert sein, dass er den Wert `hostPort` für den

Container in der Aufgabendefinition Ihres Services verwendet, die Sie bei der Zustandsprüfung verwenden.

Ping-Pfad

Dies ist Teil des Load Balancer-Healthchecks. Es handelt sich um einen Endpunkt in Ihrer Anwendung, der einen erfolgreichen Statuscode (z. B. 200) zurückgeben kann, wenn die Anwendung fehlerfrei ist. Dieser Wert wird oft auf `index.html` festgelegt, aber wenn Ihr Service auf diese Anfrage nicht antwortet, schlägt die Zustandsprüfung fehl. Wenn Ihr Container keine Datei `index.html` hat, können Sie dies auf `/` festlegen, um so auf die Basis-URL für die Container-Instance abzuzielen.

Reaktions-Timeout

Dies ist die Zeitdauer, innerhalb derer Ihr Container eine Antwort auf den Ping der Zustandsprüfung zurücksenden muss. Wenn dieser Wert niedriger ist als die Zeitdauer, die für eine Antwort erforderlich ist, schlägt die Zustandsprüfung fehl.

Zustandsprüfungsintervall

Dies ist die Zeitdauer zwischen Zustandsprüfungs-Pings. Je kürzer Ihre Zustandsprüfungsintervalle sind, desto schneller kann Ihre Container-Instance den Unhealthy Threshold erreichen.

Unhealthy Threshold (Schwellenwert für anormalen Zustand)

Dies ist die Anzahl der Male, die Ihre Zustandsprüfung fehlschlagen kann, bevor Ihre Container-Instance als fehlerhaft betrachtet wird. Wenn Sie einen Schwellenwert von 2 und ein Intervall für die Integritätsprüfung von 30 Sekunden haben, hat Ihre Aufgabe 60 Sekunden Zeit, um auf den Systemdiagnose-Ping zu antworten, bevor er als fehlerhaft eingestuft wird. Sie können den Unhealthy Threshold oder das Zustandsprüfungsintervall erhöhen, um Ihren Aufgaben mehr Zeit zum Antworten zu geben.

Der Dienstname konnte nicht aktualisiert werden: Der *Name* oder Port des Load Balancer-Containers wurde in der Aufgabendefinition geändert

Wenn Ihr Service einen Load Balancer verwendet, können Sie das AWS CLI oder SDK verwenden, um die Load Balancer-Konfiguration zu ändern. Informationen zum Ändern der Konfiguration finden Sie UpdateServicein der Amazon Elastic Container Service API-Referenz. Wenn Sie die Aufgabendefinition für den Service aktualisieren, müssen der Containername und der Container-Port, die in der Load-Balancer-Konfiguration angegeben sind, in der Aufgabendefinition verbleiben.

Sie haben das Limit für die Anzahl der Aufgaben erreicht, die Sie gleichzeitig ausführen können.

> Für ein neues Konto sind Ihre Kontingente möglicherweise niedriger als die Service Quotas. Das Servicekontingent für Ihr Konto kann in der Service-Quotas-Konsole angezeigt werden. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter Anfordern einer Kontingenterhöhung im Benutzerhandbuch zu Service Quotas.

Fehlerbehebung bei Service Auto Scaling in Amazon ECS

Application Auto Scaling deaktiviert Scale-In-Prozesse, während Amazon ECS-Bereitstellungen laufen, und sie werden fortgesetzt, sobald die Bereitstellung abgeschlossen ist. Scale-Out-Prozesse werden während einer Bereitstellung jedoch weiterhin ausgeführt, es sei denn, sie werden angehalten. Weitere Informationen finden Sie unter Unterbrechen und Wiederaufnehmen der Skalierung für Application Auto Scaling.

Problembehandlung bei ungültigen CPU- oder Speicherfehlern mit Amazon ECS-Aufgabendefinition

Bei der Registrierung einer Aufgabendefinition mithilfe der Amazon ECS-API oder AWS CLI, wenn Sie einen ungültigen `cpu` `memory` Wert angeben, wird der folgende Fehler zurückgegeben.

```
An error occurred (ClientException) when calling the RegisterTaskDefinition operation:
 Invalid 'cpu' setting for task.
```

> **Note**
>
> Bei der Verwendung von Terraform wird möglicherweise der folgende Fehler zurückgegeben.
>
> ```
> Error: ClientException: No Fargate configuration exists for given values.
> ```

Um dieses Problem zu lösen, müssen Sie in Ihrer Aufgabendefinition einen unterstützten Wert für die Aufgaben-CPU und den Speicher angeben. Der `cpu` Wert kann in einer Aufgabendefinition in CPU-Einheiten oder vCPUs ausgedrückt werden. Er wird in eine Ganzzahl umgewandelt, die die CPU-Einheiten angibt, wenn die Aufgabendefinition registriert wird. Der `memory` Wert kann in einer

Aufgabendefinition in MiB oder GB ausgedrückt werden. Es wird in eine Ganzzahl umgewandelt, die die MiB angibt, wenn die Aufgabendefinition registriert wird.

Für Aufgabendefinitionen, die nur `EC2` für den `requiresCompatibilities`-Parameter angeben, liegen die unterstützten CPU-Werte zwischen `256` CPU-Einheiten (`0.25` vCPUs) und `16384` CPU-Einheiten (16 vCPUs). Der Speicherwert muss eine Ganzzahl sein, und das Limit hängt von der Menge des verfügbaren Speichers auf der zugrunde liegenden Amazon EC2 EC2-Instance ab, die Sie verwenden.

Für Aufgabendefinitionen, die `FARGATE` für den `requiresCompatibilities` Parameter spezifizieren (auch wenn dieser ebenfalls angegeben `EC2` ist), müssen Sie einen der Werte in der folgenden Tabelle verwenden. Diese Werte bestimmen den Bereich der unterstützten Werte für den CPU- und Speicherparameter.

Für Aufgaben, die auf Fargate gehostet werden, zeigt die folgende Tabelle die gültigen CPU- und Arbeitsspeicher-Kombinationen. Die Speicherwerte in der JSON-Datei sind in MiB angegeben. Sie können den GB-Wert in MiB konvertieren, indem Sie den Wert mit 1 024 multiplizieren. Zum Beispiel 1 GB = 1 024 MiB.

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
256 (0,25 vCPU)	512 MiB, 1 GB, 2 GB	Linux
512 (0,5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	Linux
1024 (1 vCPU)	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB	Linux, Windows
2048 (2 vCPU)	Zwischen 4 GB und 16 GB in 1-GB-Schritten	Linux, Windows
4096 (4 vCPU)	Zwischen 8 GB und 30 GB in 1-GB-Schritten	Linux, Windows
8 192 (8 vCPU)	Zwischen 16 GB und 60 GB in 4-GB-Schritten	Linux

CPU-Wert	Speicherwert	Für AWS Fargate unterstützte Betriebssysteme
Note Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.		
16 384 (16 vCPU) **Note** Diese Option erfordert die Linux-Plattform `1.4.0` oder höher.	Zwischen 32 GB und 120 GB in 8-GB-Schritten	Linux

Für Aufgaben, die auf Amazon EC2 gehostet werden, liegen die unterstützten Task-CPU-Werte zwischen 0,25 vCPUs und 192 vCPUs.

> **Note**
>
> CPU- und Speicherparameter auf Aufgabenebene werden für Windows-Container ignoriert.

Amazon ECS-Container-Agent-Protokolle anzeigen

Amazon ECS speichert Protokolle im Ordner `/var/log/ecs` Ihrer Container-Instances. Es sind Protokolle vom Amazon-ECS-Container-Agenten und vom `ecs-init`-Service verfügbar, der den Status des Agenten (Start/Stopp) auf der Container-Instance kontrolliert. Sie können diese Protokolldateien anzeigen, indem Sie sich mithilfe von SSH mit der Container-Instance verbinden.

> **Note**
>
> Wenn Sie nicht sicher sind, wie Sie alle verschiedenen Protokolle auf Ihren Container-Instances sammeln können, können Sie den Amazon-ECS-Protokollsammler verwenden.

Weitere Informationen finden Sie unter Sammeln von Container-Protokollen mit Amazon ECS Logs Collector .

Linux Operating System

Der Prozess `ecs-init` speichert Protokolle unter `/var/log/ecs/ecs-init.log`.

Die `ecs-init.log` Datei enthält Informationen zum Lebenszyklusmanagement, zur Konfiguration und zum Bootstrapping des Continer-Agenten.

Sie können den folgenden Befehl verwenden, um die Protokolldateien anzuzeigen.

```
cat /var/log/ecs/ecs-init.log
```

Ausgabe:

```
2018-02-16T18:13:54Z [INFO] pre-start
2018-02-16T18:13:56Z [INFO] start
2018-02-16T18:13:56Z [INFO] No existing agent container to remove.
2018-02-16T18:13:56Z [INFO] Starting Amazon Elastic Container Service Agent
```

Windows Operating System

Sie können den Amazon ECS-Protokollsammler für Windows verwenden. Weitere Informationen finden Sie unter Amazon ECS Logs Collector für Windows auf Github.

1. Verbinden Sie sich mit der Instance.
2. Öffnen Sie die folgenden Befehle PowerShell und führen Sie sie dann mit Administratorrechten aus. Die Befehle laden das Skript herunter und sammeln die Protokolle.

   ```
   Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
   raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
   logs-collector.ps1
   .\ecs-logs-collector.ps1
   ```

Sie können die Debug-Protokollierung für den Amazon ECS-Agenten und den Docker-Daemon aktivieren. Diese Option ermöglicht es dem Skript, die Protokolle zu sammeln, bevor der Debug-

Modus aktiviert wird. Das Skript startet den Docker-Daemon und den Amazon ECS-Agenten neu und beendet dann alle Container, die auf der Instance ausgeführt werden. Bevor Sie den folgenden Befehl ausführen, leeren Sie die Container-Instance und verschieben Sie alle wichtigen Aufgaben auf andere Container-Instances.

Führen Sie den folgenden Befehl aus, um die Protokollierung zu aktivieren.

```
.\ecs-logs-collector.ps1 -RunMode debug
```

Sammeln von Container-Protokollen mit Amazon ECS Logs Collector

Wenn Sie nicht sicher sind, wie Sie all die verschiedenen Protokolle auf Ihren Container-Instances sammeln können, können Sie den Amazon-ECS-Protokollsammler verwenden. Er ist sowohl GitHub für Linux als auch für Windows verfügbar. Das Skript sammelt allgemeine Betriebssystemprotokolle sowie Docker- und Amazon ECS-Container-Agent-Protokolle, die bei der Fehlerbehebung hilfreich sein können AWS Support . Anschließend komprimiert und archiviert es die erfassten Informationen in einer einzigen Datei, die problemlos für Diagnosezwecke weitergegeben werden kann. Außerdem unterstützt es die Aktivierung des Debug-Modus für den Docker-Daemon und den Amazon-ECS-Container-Agenten auf Amazon Linux-Varianten, wie etwa das Amazon-ECS-optimierte AMI. Derzeit unterstützt der Amazon-ECS-Protokollsammler die folgenden Betriebssysteme:

- Amazon Linux
- Red Hat Enterprise Linux 7
- Debian 8
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.04
- Windows Server 2016

> **Note**
>
> Der Quellcode für den Amazon ECS Logs Collector ist sowohl GitHub für Linux als auch für Windows verfügbar. Wir möchten Sie bitten, uns eventuelle Änderungswünsche mitzuteilen.

Amazon Web Services bietet derzeit jedoch keine Unterstützung für die Ausführung modifizierter Kopien dieser Software.

So laden Sie den Amazon-ECS-Protokollsammler für Linux herunter und führen ihn aus

1. Stellen Sie eine Verbindung mit Ihrer Container-Instance her.
2. Laden Sie das Skript des Amazon-ECS-Protokollsammlers herunter.

```
curl -O https://raw.githubusercontent.com/awslabs/ecs-logs-collector/master/ecs-logs-collector.sh
```

3. Führen Sie das Skript aus, um die Protokolle zu erfassen und das Archiv zu erstellen.

> **Note**
>
> Um den Debug-Modus für den Docker-Daemon und den Amazon ECS-Container-Agenten zu aktivieren, fügen Sie die `--mode=enable-debug` Option zum folgenden Befehl hinzu. Dadurch könnte der Docker-Daemon neu gestartet werden, der alle Container beendet, die auf der Instance ausgeführt werden. Sie sollten in Betracht ziehen, die Container-Instance auszugleichen und alle wichtigen Aufgaben in andere Container-Instances zu verschieben, bevor Sie den Debug-Modus aktivieren. Weitere Informationen finden Sie unter Entleeren von Amazon ECS-Container-Instances.

```
[ec2-user ~]$ sudo bash ./ecs-logs-collector.sh
```

Nachdem Sie das Skript ausgeführt haben, können Sie die gesammelten Protokolle im Ordner `collect`, den das Skript erstellt hat, untersuchen. Bei der `collect.tgz` Datei handelt es sich um ein komprimiertes Archiv mit allen Protokollen, das Sie AWS Support zur Diagnoseunterstützung zur Verfügung stellen können.

So laden Sie den Amazon-ECS-Protokollsammler für Windows herunter und führen ihn aus

1. Stellen Sie eine Verbindung mit Ihrer Container-Instance her. Weitere Informationen finden Sie unter Connecting to Your Windows Instance im Amazon EC2 EC2-Benutzerhandbuch.
2. Laden Sie das Amazon ECS-Logs-Collector-Skript mit herunter PowerShell.

```
Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
logs-collector.ps1
```

3. Führen Sie das Skript aus, um die Protokolle zu erfassen und das Archiv zu erstellen.

> **Note**
>
> Um den Debug-Modus für den Docker-Daemon und den Amazon ECS-Container-Agenten zu aktivieren, fügen Sie die `-RunMode debug` Option zum folgenden Befehl hinzu. Dies startet den Docker-Daemon neu, wodurch alle aktuell auf der Instance ausgeführten Container gestoppt werden. Sie sollten in Betracht ziehen, die Container-Instance auszugleichen und alle wichtigen Aufgaben in andere Container-Instances zu verschieben, bevor Sie den Debug-Modus aktivieren. Weitere Informationen finden Sie unter Entleeren von Amazon ECS-Container-Instances.

```
.\ecs-logs-collector.ps1
```

Nachdem Sie das Skript ausgeführt haben, können Sie die gesammelten Protokolle im Ordner `collect`, den das Skript erstellt hat, untersuchen. Bei der `collect.tgz` Datei handelt es sich um ein komprimiertes Archiv aller Protokolle, das Sie mit dem AWS Support teilen können, um Hilfe bei der Diagnose zu erhalten.

Rufen Sie Amazon ECS-Diagnosedetails mit Agenten-Introspektion ab

Die Amazon ECS Agent Introspection-API bietet Informationen über den Gesamtstatus des Amazon ECS-Agenten und der Container-Instances.

Sie können die Agent-Introspection-API verwenden, um die Docker-ID für einen Container in Ihrer Aufgabe abzurufen. Sie können die Agenten-Introspektions-API verwenden, indem Sie sich mithilfe von SSH mit einer Container-Instance verbinden.

> **Important**
>
> Ihre Container-Instance muss über eine IAM-Rolle verfügen, die den Zugriff auf Amazon ECS erlaubt, um die Introspektions-API zu erreichen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

Das folgende Beispiel zeigt zwei Aufgaben, eine, die gerade ausgeführt wird, und eine, die gestoppt wurde.

> **Note**
>
> Der folgende Befehl wird aus Gründen der besseren Lesbarkeit über die python -mjson.tool weitergeleitet.

```
curl http://localhost:51678/v1/tasks | python -mjson.tool
```

Ausgabe:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1095  100  1095    0     0   117k      0 --:--:-- --:--:-- --:--:--  133k
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/090eff9b-1ce3-4db6-848a-
a8d14064fd24",
            "Containers": [
                {
                    "DockerId":
 "189a8ff4b5f04affe40e5160a5ffadca395136eb5faf4950c57963c06f82c76d",
                    "DockerName": "ecs-console-sample-app-static-6-simple-
app-86caf9bcabe3e9c61600",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "f7f1f8a7a245c5da83aa92729bd28c6bcb004d1f6a35409e4207e1d34030e966",
                    "DockerName": "ecs-console-sample-app-static-6-busybox-
ce83ce978a87a890ab01",
```

```
                    "Name": "busybox"
                }
            ],
            "Family": "console-sample-app-static",
            "KnownStatus": "STOPPED",
            "Version": "6"
        },
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/1810e302-eaea-4da9-
a638-097bea534740",
            "Containers": [
                {
                    "DockerId":
 "dc7240fe892ab233dbbcee5044d95e1456c120dba9a6b56ec513da45c38e3aeb",
                    "DockerName": "ecs-console-sample-app-static-6-simple-app-
f0e5859699a7aecfb101",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "096d685fb85a1ff3e021c8254672ab8497e3c13986b9cf005cbae9460b7b901e",
                    "DockerName": "ecs-console-sample-app-static-6-
busybox-92e4b8d0ecd0cce69a01",
                    "Name": "busybox"
                }
            ],
            "DesiredStatus": "RUNNING",
            "Family": "console-sample-app-static",
            "KnownStatus": "RUNNING",
            "Version": "6"
        }
    ]
}
```

Im vorherigen Beispiel hat die gestoppte Aufgabe (*090eff9b-1ce3-4db6-848a-a8d14064fd24*) zwei Container. Mit docker inspect ***container-ID*** können Sie detaillierte Informationen zu jedem Container anzeigen. Weitere Informationen finden Sie unter Introspektion von Amazon ECS-Containern.

Docker-Diagnose in Amazon ECS

Docker bietet mehrere Diagnosetools, die bei der Behebung von Problemen mit Ihren Containern und Aufgaben helfen. Weitere Informationen über alle verfügbaren Docker-Befehlszeilen-Tools finden Sie im Thema Docker Command Line in der Docker-Dokumentation. Sie können auf die Docker-Befehlszeilen-Tools zugreifen, indem Sie mithilfe von SSH eine Verbindung zu einer Container-Instance herstellen.

Die Exitcodes, die Docker-Container berichten, können auch Diagnoseinformationen enthalten (zum Beispiel bedeutet Exitcode 137, dass der Container ein Signal `SIGKILL` empfangen hat). Weitere Informationen finden Sie unter Exit Status in der Docker-Dokumentation.

Docker-Container in Amazon ECS auflisten

Sie können mithilfe des Befehls docker ps auf Ihrer Container-Instance die Container auflisten, die gerade ausgeführt werden. Im folgenden Beispiel läuft nur der Amazon ECS-Container-Agent. Weitere Informationen finden Sie unter docker ps in der Docker-Dokumentation.

```
docker ps
```

Ausgabe:

```
CONTAINER ID        IMAGE                            COMMAND             CREATED
      STATUS              PORTS                          NAMES
cee0d6986de0        amazon/amazon-ecs-agent:latest   "/agent"            22 hours ago
      Up 22 hours         127.0.0.1:51678->51678/tcp   ecs-agent
```

Sie können mithilfe des Befehls docker ps -a alle Container anzeigen (auch gestoppte oder abgebrochene Container). Dies ist hilfreich, um Container aufzulisten, die unerwartet gestoppt wurden. Im folgenden Beispiel wurde der Container `f7f1f8a7a245` vor 9 Sekunden beendet, daher wird er in einer docker ps-Ausgabe ohne das Flag `-a` nicht aufgeführt.

```
docker ps -a
```

Ausgabe:

```
CONTAINER ID        IMAGE                                        COMMAND
 CREATED             STATUS                      PORTS                         NAMES
```

```
db4d48e411b1        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  19 seconds ago                                                          ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-c09288a6b0cba8a53700
f7f1f8a7a245        busybox:buildroot-2014.02                   "\"sh -c '/bin/sh -c
  22 hours ago        Exited (137) 9 seconds ago                          ecs-
console-sample-app-static-6-busybox-ce83ce978a87a890ab01
189a8ff4b5f0        httpd:2                                     "httpd-foreground"
  22 hours ago        Exited (137) 40 seconds ago                         ecs-
console-sample-app-static-6-simple-app-86caf9bcabe3e9c61600
0c7dca9321e3        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  22 hours ago                                                            ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-90fefaa68498a8a80700
cee0d6986de0        amazon/amazon-ecs-agent:latest              "/agent"
  22 hours ago        Up 22 hours                 127.0.0.1:51678->51678/tcp   ecs-
agent
```

Docker-Protokolle in Amazon ECS anzeigen

Sie können die Streams `STDOUT` und `STDERR` für einen Container mithilfe des Befehls docker logs anzeigen. In diesem Beispiel werden die Protokolle für den Container *`dc7240fe892a`* angezeigt und per Pipe durch den Befehl head geleitet, um die Ausgabe zu kürzen. Weitere Informationen finden Sie unter docker logs in der Docker-Dokumentation.

> **Note**
>
> Docker-Protokolle stehen nur in der Container-Instance zur Verfügung, wenn Sie den Standardtreiber für `json`-Protokolle verwenden. Wenn Sie Ihre Aufgaben für die Verwendung des `awslogs` Protokolltreibers konfiguriert haben, sind Ihre Container-Logs unter CloudWatch Logs verfügbar. Weitere Informationen finden Sie unter Amazon ECS-Protokolle senden an CloudWatch .

```
docker logs dc7240fe892a | head
```

Ausgabe:

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
```

```
[Thu Apr 23 19:48:36.956682 2015] [mpm_event:notice] [pid 1:tid 140327115417472]
 AH00489: Apache/2.4.12 (Unix) configured -- resuming normal operations
[Thu Apr 23 19:48:36.956827 2015] [core:notice] [pid 1:tid 140327115417472] AH00094:
 Command line: 'httpd -D FOREGROUND'
10.0.1.86 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:29 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.0.154 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.1.86 - - [23/Apr/2015:19:49:58 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:50:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:50:29 +0000] "GET / HTTP/1.1" 200 348
time="2015-04-23T20:11:20Z" level="fatal" msg="write /dev/stdout: broken pipe"
```

Untersuchen Sie Docker-Container in Amazon ECS

Wenn Sie die Docker-ID eines Containers haben, können Sie ihn mithilfe des Befehls docker inspect untersuchen. Die Untersuchung von Containern bietet die detaillierteste Ansicht der Umgebung, in der der Container gestartet wurde. Weitere Informationen finden Sie unter docker inspect in der Docker-Dokumentation.

```
docker inspect dc7240fe892a
```

Ausgabe:

```
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": false,
        "AttachStdin": false,
        "AttachStdout": false,
        "Cmd": [
            "httpd-foreground"
        ],
        "CpuShares": 10,
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": [
```

```
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/apache2/bin",
            "HTTPD_PREFIX=/usr/local/apache2",
            "HTTPD_VERSION=2.4.12",
            "HTTPD_BZ2_URL=https://www.apache.org/dist/httpd/httpd-2.4.12.tar.bz2"
        ],
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Hostname": "dc7240fe892a",
...
```

Konfiguration der ausführlichen Ausgabe vom Docker-Daemon in Amazon ECS

Wenn Sie Probleme mit Docker-Containern oder -Images haben, können Sie den Debug-Modus auf Ihrem Docker-Daemon aktivieren. Die Verwendung von Debugging bietet eine ausführlichere Ausgabe vom Daemon. Sie können damit Fehlermeldungen abrufen, die von Container-Registern wie Amazon ECR gesendet werden.

> **Important**
>
> Dieses Verfahren wurde für das Amazon-ECS-optimierte Amazon Linux AMI geschrieben. Informationen zu anderen Betriebssystemen finden Sie unter Enable Debugging and Control and Configure with in der Docker-Dokumentation. Docker systemd

So verwenden Sie den Docker-Daemon-Debug-Modus auf dem Amazon ECS-optimierten Amazon Linux AMI

1. Stellen Sie eine Verbindung mit Ihrer Container-Instance her.
2. Öffnen Sie die Datei mit den Docker-Optionen in einem Text-Editor, wie z. B. vi. Für das Amazon-ECS-optimierte Amazon Linux AMI befindet sich die Datei mit den Docker-Optionen unter `/etc/sysconfig/docker`.
3. Suchen Sie nach der Anweisung für die Docker-Optionen und fügen Sie die Option `-D` in Anführungszeichen zur Zeichenfolge hinzu.

> **Note**
>
> Wenn die Anweisung für die Docker-Optionen mit einem # beginnt, entfernen Sie dieses Zeichen, um die Auskommentierung der Anweisung aufzuheben und die Optionen zu aktivieren.

Für das Amazon-ECS-optimierte Amazon Linux AMI wird die Anweisung für die Docker-Optionen `OPTIONS` genannt. Zum Beispiel:

```
# Additional startup options for the Docker daemon, for example:
# OPTIONS="--ip-forward=true --iptables=true"
# By default we limit the number of open files per container
OPTIONS="-D --default-ulimit nofile=1024:4096"
```

4. Speichern Sie die Datei und beenden Sie den Text-Editor.
5. Starten Sie den Docker-Daemon erneut.

```
sudo service docker restart
```

Die Ausgabe sieht wie folgt aus:

```
Stopping docker:                                           [  OK  ]
Starting docker: .                                    [  OK  ]
```

6. Starten Sie den Amazon-ECS-Agenten neu.

```
sudo service ecs restart
```

Ihre Docker-Protokolle sollten jetzt mehr Informationen enthalten.

```
time="2015-12-30T21:48:21.907640838Z" level=debug msg="Unexpected response from
 server: \"{\\\"errors\\\":[{\\\"code\\\":\\\"DENIED\\\",\\\"message\\\":\\\"User:
 arn:aws:sts::1111:assumed-role/ecrReadOnly/i-abcdefg is not authorized to perform:
 ecr:InitiateLayerUpload on resource: arn:aws:ecr:us-east-1:1111:repository/nginx_test
\\\"}]}\\n\" http.Header{\"Connection\":[]string{\"keep-alive\"}, \"Content-Type\":
[]string{\"application/json; charset=utf-8\"}, \"Date\":[]string{\"Wed, 30 Dec 2015
```

```
21:48:21 GMT\"}, \"Docker-Distribution-Api-Version\":[]string{\"registry/2.0\"},
\"Content-Length\":[]string{\"235\"}}"
```

Fehlerbehebung beim Docker **`API error (500): devmapper`** in Amazon ECS

Der folgende Docker-Fehler gibt an, dass der Thin-Pool-Speicher für Ihre Container-Instance vollständig belegt ist und der Docker-Daemon keine neuen Container erstellen kann:

```
CannotCreateContainerError: API error (500): devmapper: Thin Pool has 4350 free data
 blocks which is less than minimum required 4454 free data blocks. Create more free
 space in thin pool or use dm.min_free_space option to change behavior
```

Amazon-ECS-optimierte Amazon Linux AMIs ab Version `2015.09.d` werden standardmäßig mit einem 8-GiB-Volume für das Betriebssystem gestartet, das unter `/dev/xvda` angefügt und als Stamm des Dateisystems gemountet ist. Es gibt ein zusätzliches 22-GiB-Volume, das unter `/dev/xvdcz` angefügt ist und das von Docker zum Speichern von Images und Metadaten verwendet wird. Wenn dieser Speicherplatz voll ist, kann der Docker-Daemon keine neuen Container mehr erstellen.

Der einfachste Weg, Speicher zu Container-Instances hinzuzufügen, besteht darin, die bestehenden Instances zu beenden und neue mit größeren Datenspeichervolumen zu starten. Wenn Sie das allerdings nicht tun können, können Sie Speicher zu der Volume-Gruppe hinzufügen, die Docker verwendet, und sein logisches Volume mithilfe der Verfahren in Amazon ECS-optimierte Linux-AMIs erweitern.

Wenn sich der Speicher Ihrer Container-Instance zu schnell füllt, können Sie einige Maßnahmen ergreifen, um diesen Effekt zu reduzieren:

- Führen Sie den folgenden Befehl für Ihre Container-Instance aus, um die Thin Poll-Informationen anzuzeigen:

  ```
  docker info
  ```

- (Amazon ECS Container Agent 1.8.0 und höher) Sie können die Zeit reduzieren, für die gestoppte oder verlassene Container auf Ihren Container-Instances verbleiben. Die Variable zur Agentenkonfiguration `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` stellt die Zeitdauer ein, die gewartet wird, von dem Zeitpunkt, zu dem eine Aufgabe gestoppt wird, bis zu dem Zeitpunkt, zu dem der Docker-Container entfernt wird (standardmäßig beträgt dieser Wert 3 Stunden). Dadurch

werden die Daten des Docker-Containers entfernt. Wenn dieser Wert zu niedrig eingestellt ist, können Sie Ihre gestoppten Container möglicherweise nicht überprüfen oder die Protokolle nicht einsehen, bevor sie entfernt werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

- Sie können nicht laufende Container und unbenutzte Images aus Ihren Container-Instances entfernen. Sie können mithilfe der folgenden Beispielbefehle gestoppte Container und nicht verwendete Images manuell löschen. Gelöschte Container können später nicht geprüft werden und gelöschte Images müssen noch mal abgerufen werden, bevor neue Container von ihnen gestartet werden können.

  Um nicht ausgeführte Container zu entfernen, führen Sie den folgenden Befehl auf Ihrer Container-Instance aus:

  ```
  docker rm $(docker ps -aq)
  ```

  Um nicht verwendete Images zu entfernen, führen Sie den folgenden Befehl auf Ihrer Container-Instance aus:

  ```
  docker rmi $(docker images -q)
  ```

- Sie können ungenutzte Datenblöcke innerhalb von Containern entfernen. Mithilfe des folgenden Befehls können Sie fstrim auf jedem laufenden Container ausführen und alle Datenblöcke verwerfen, die von dem Container-Dateisystem nicht verwendet werden.

  ```
  sudo sh -c "docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -IZ fstrim /proc/Z/root/"
  ```

Probleme mit Amazon ECS Exec beheben

Im Folgenden finden Sie Hinweise zur Fehlerbehebung, um zu diagnostizieren, warum Sie bei der Verwendung von ECS Exec einen Fehler erhalten.

Überprüfen Sie dies mit dem Exec Checker

Mit dem ECS Exec Checker-Skript können Sie überprüfen und validieren, ob Ihr Amazon ECS-Cluster und Ihre Aufgabe die Voraussetzungen für die Verwendung der ECS Exec-Funktion erfüllen. Das ECS Exec Checker-Skript verifiziert, dass sowohl Ihre AWS CLI Umgebung als auch Ihr

Cluster und die Aufgaben für ECS Exec bereit sind, indem es in Ihrem Namen verschiedene APIs aufruft. Das Tool benötigt die neueste Version von AWS CLI und muss verfügbar sein. `jq` Weitere Informationen finden Sie unter ECS Exec Checker unter. GitHub

Fehler beim Aufruf von **`execute-command`**

Wenn ein `The execute command failed`-Fehler auftritt, könnte folgendes die möglichen Ursachen sein.

- Der Task verfügt nicht über die erforderlichen Berechtigungen. Stellen Sie sicher, dass für die Aufgabendefinition, die zum Starten der Aufgabe verwendet wird, eine Aufgaben-IAM-Rolle definiert ist und dass die Rolle über die erforderlichen Berechtigungen verfügt. Weitere Informationen finden Sie unter ECS Exec-Berechtigungen.
- Der SSM-Agent ist nicht installiert oder läuft nicht.
- Es gibt eine Amazon VPC-Schnittstelle für Amazon ECS, aber keine für Systems Manager Session Manager.

Probleme mit Amazon ECS Anywhere beheben

Amazon ECS Anywhere bietet Unterstützung für die Registrierung einer externen Instanz, z. B. eines lokalen Servers oder einer virtuellen Maschine (VM), in Ihrem Amazon ECS-Cluster. Im Folgenden finden Sie häufig auftretende Probleme sowie allgemeine Empfehlungen zur Fehlerbehebung für sie.

Themen

Registrierungsprobleme bei externen Instances

Wenn Sie eine externe Instance bei Ihrem Amazon-ECS-Cluster registrieren, müssen folgende Anforderungen erfüllt sein:

- Eine AWS Systems Manager Aktivierung, die aus einer Aktivierungs-ID und einem Aktivierungscode besteht, muss abgerufen werden. Sie verwenden sie, um die externe Instance als verwaltete Instance von Systems Manager zu registrieren. Wenn eine Systems Manager Manager-

Aktivierung angefordert wird, geben Sie ein Registrierungslimit und ein Ablaufdatum an. Die Registrierungsgrenze gibt die maximale Anzahl der Instances an, die mit der Aktivierung registriert werden können. Der Standardwert für das Registrierungslimit ist 1 Instanz. Das Ablaufdatum gibt an, wann die Aktivierung abläuft. Der Standardwert beträgt 24 Stunden. Wenn die Systems Manager-Aktivierung, die Sie zur Registrierung Ihrer externen Instance verwenden, nicht gültig ist, fordern Sie eine neue an. Weitere Informationen finden Sie unter Registrierung einer externen Instance in einem Amazon ECS-Cluster.

- Eine IAM-Richtlinie wird verwendet, um Ihrer externen Instance die Berechtigungen zu gewähren, die sie für die Kommunikation mit AWS API-Vorgängen benötigt. Wenn diese verwaltete Richtlinie nicht ordnungsgemäß erstellt wurde und nicht die erforderlichen Berechtigungen enthält, schlägt die Registrierung externer Instances fehl. Weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere.
- Amazon ECS stellt ein Installationsskript bereit, das Docker, den Amazon-ECS-Container Agent und den Systems Manager Agent auf Ihrer externen Instance installiert. Wenn das Installationsskript fehlschlägt, kann das Skript wahrscheinlich nicht erneut auf derselben Instance ausgeführt werden, ohne dass ein Fehler auftritt. Folgen Sie in diesem Fall dem Bereinigungsprozess, um AWS Ressourcen aus der Instanz zu löschen, sodass Sie das Installationsskript erneut ausführen können. Weitere Informationen finden Sie unter Deregistrierung einer externen Amazon ECS-Instance.

  > **Note**
  >
  > Beachten Sie, dass, wenn das Installationsskript die Aktivierung von Systems Manager erfolgreich angefordert und verwendet hat, die Aktivierung des Systems Managers erneut verwendet wird, wenn das Installationsskript ein zweites Mal ausgeführt wird. Dies kann wiederum dazu führen, dass Sie das Registrierungslimit für diese Aktivierung erreichen. Wenn diese Limits erreicht ist, müssen Sie eine neue Aktivierung erstellen.

- Wenn das Installationsskript auf einer externen Instanz für GPU-Workloads ausgeführt wird und der NVIDIA-Treiber nicht richtig erkannt oder konfiguriert wird, tritt ein Fehler auf. Das Installationsskript verwendet den `nvidia-smi`-Befehl, um das Vorhandensein des NVIDIA-Treibers zu bestätigen.

Netzwerkprobleme mit externen Instances

Um Änderungen mitzuteilen, benötigt Ihre externe Instance eine Netzwerkverbindung zu AWS. Wenn Ihre externe Instance ihre Netzwerkverbindung zu verliert, werden Aufgaben AWS, die auf Ihren

Instances ausgeführt werden, trotzdem weiter ausgeführt, sofern sie nicht manuell beendet werden. Nachdem die Verbindung zu wiederhergestellt AWS ist, werden die AWS Anmeldeinformationen, die vom Amazon ECS-Container-Agenten und dem Systems Manager Manager-Agenten auf der externen Instance verwendet werden, automatisch erneuert. Weitere Informationen zu den AWS Domänen, die für die Kommunikation zwischen Ihrer externen Instance und verwendet werden AWS, finden Sie unterNetzwerk .

Probleme beim Ausführen von Aufgaben auf Ihrer externen Instance

Wenn Ihre Aufgaben oder Container auf Ihrer externen Instance nicht ausgeführt werden können, sind die häufigsten Ursachen entweder netzwerk- oder berechtigungsbezogen. Wenn Ihre Container ihre Images von Amazon ECR abrufen oder so konfiguriert sind, dass sie Container-Logs an CloudWatch Logs senden, muss Ihre Aufgabendefinition eine gültige IAM-Rolle für die Aufgabenausführung angeben. Ohne eine gültige IAM-Rolle für die Aufgabenausführung können Ihre Container nicht gestartet werden. Weitere Informationen zu Netzwerkproblemen finden Sie unter Netzwerkprobleme mit externen Instances.

> **Important**
>
> Amazon ECS bietet das Tool zur Erfassung von Amazon-ECS-Protokollen. Sie können es verwenden, um Protokolle von Ihren externen Instances zu Fehlerbehebungszwecken zu sammeln. Weitere Informationen finden Sie unter Sammeln von Container-Protokollen mit Amazon ECS Logs Collector .

AWS Fargate Drosselung der Kontingente

AWS Fargate begrenzt die Startraten für Amazon ECS-Aufgaben und Amazon EKS-Pods auf Kontingente (früher als Limits bezeichnet). Dabei wird ein Token-Bucket-Algorithmus für jedes AWS Konto pro Region verwendet. Mit diesem Algorithmus verfügt Ihr Konto über einen Bucket, der eine bestimmte Anzahl von Tokens enthält. Die Anzahl der Tokens im Bucket entspricht Ihrem Ratenkontingent zu einer bestimmten Sekunde. Jedes Kundenkonto verfügt über einen Token-Bucket für Aufgaben und Pods, der basierend auf der Anzahl der Aufgaben und Pods, die vom Kundenkonto gelauncht werden, erschöpft wird. Dieser Token-Bucket verfügt über ein Bucket-Maximum, mit dem Sie regelmäßig eine höhere Anzahl von Anforderungen stellen können, und eine Nachfüllrate, mit der Sie eine stetige Rate von Anfragen so lange wie nötig aufrechterhalten können.

Zum Beispiel beträgt die Bucket-Größe für Aufgaben und Pods für ein Fargate-Kundenkonto 100 Token und die Nachfüllrate beträgt 20 Token pro Sekunde. Daher können Sie sofort bis zu 100 Amazon-ECS-Aufgaben und Amazon-EKS-Pods pro Kundenkonto launchen, mit einer anhaltenden Launch-Rate von 20 Amazon-ECS-Aufgaben und Amazon-EKS-Pods pro Sekunde.

Aktionen	Maximale Kapazität des Buckets (oder Burst-Rate)	Nachfüllrate des Buckets (oder konstante Rate)
Ratenkontingent für Fargate-Ressourcen für On-Demand-Amazon-ECS-Aufgaben und Amazon-EKS-Pods[1]	100	20
Ratenkontingent für Fargate-Ressourcen für Spot-Amazon-ECS-Aufgaben	100	20

[1]Konten, die nur Amazon-EKS-Pods launchen, haben eine Burst-Rate von 20 mit einer konstanten Pod-Launch-Rate von 20 Pod-Launches pro Sekunde, wenn sie die in den Amazon-EKS-Plattformversionen aufgerufenen Plattformversionen verwenden.

Drosselung der **RunTask** API in Fargate

Darüber hinaus begrenzt Fargate die Anforderungsrate beim Launchen von Aufgaben mit der `RunTask`-API von Amazon ECS unter einem separaten Kontingent. Fargate begrenzt Amazon `RunTask` ECS-API-Anfragen für jedes AWS Konto auf regionaler Basis. Jede Anforderung, die Sie stellen, entfernt ein Token aus dem Bucket. Wir tun dies, um die Leistung des Dienstes zu unterstützen und um eine faire Nutzung für alle Fargate-Kunden sicherzustellen. API-Aufrufe unterliegen den Anforderungs-Kontingenten, unabhängig davon, ob sie von der Konsole von Amazon Elastic Container Service, einem Befehlszeilen-Tool oder einer Anwendung eines Drittanbieters stammen. Das Ratenkontingent für Aufrufe an die `RunTask`-API von Amazon ECS besteht aus 20 Aufrufen pro Sekunde (Burst und konstant). Jeder Aufruf an diese API kann jedoch bis zu 10 Aufgaben launchen. Das bedeutet, dass Sie 100 Aufgaben in einer Sekunde launchen können, indem Sie 10 Aufrufe an diese API tätigen und bei jedem Aufruf den Launch von 10 Aufgaben anfordern. In ähnlicher Weise könnten Sie auch 20 Aufrufe an diese API tätigen und bei jedem Aufruf den Launch von 5 Aufgaben anfordern. Weitere Informationen zur API-Drosselung für die Amazon `RunTask` ECS-API finden Sie unter API-Anforderungsdrosselung in der Amazon ECS-API-Referenz.

In der Praxis hängen die Launch-Raten von Aufgaben und Pods auch von anderen Überlegungen ab, wie zum Beispiel herunterzuladende und zu entpackende Container-Images, Zustandsprüfungen und andere aktivierte Integrationen, wie das Registrieren von Aufgaben oder Pods in einem Load Balancer. Kunden stellen fest, dass die Startraten für Aufgaben und Pods im Vergleich zu den zuvor angegebenen Kontingenten variieren, je nachdem, welche Funktionen die Kunden aktivieren.

Anpassung der Zollkontingente in Fargate

Sie können eine Erhöhung der Raten-Drosselungs-Kontingente für Fargate für Ihr AWS -Konto anfordern. Um eine Kontingentanpassung anzufordern, kontaktieren Sie das AWS Support -Center.

Probleme mit der Drosselung von Amazon ECS lösen

Drosselungsfehler lassen sich in zwei Hauptkategorien einteilen: synchrone Drosselung und asynchrone Drosselung.

Synchrone Drosselung

Wenn synchrone Drosselung auftritt, erhalten Sie sofort eine Fehlerantwort von Amazon ECS. Diese Kategorie der Drosselung tritt normalerweise auf, wenn Sie Amazon ECS-APIs aufrufen, während Sie Aufgaben ausführen oder Dienste erstellen. Weitere Informationen über die damit verbundene Drosselung und die entsprechenden Drosselungsgrenzen finden Sie unter Drosselung von Anfragen für die Amazon ECS-API.

Wenn Ihre Anwendung API-Anfragen initiiert, beispielsweise mithilfe des AWS CLI oder eines AWS SDK, können Sie die API-Drosselung beheben. Sie können dies tun, indem Sie entweder Ihre Anwendung so gestalten, dass sie die Fehler behandelt, oder indem Sie eine exponentielle Backoff- und Jitter-Strategie mit Wiederholungslogik für die API-Aufrufe implementieren. Weitere Informationen finden Sie unter Timeouts, Wiederholungen und Backoff mit Jitter.

Wenn Sie ein AWS SDK verwenden, ist die automatische Wiederholungslogik bereits integriert und konfigurierbar.

Asynchrone Drosselung in Amazon ECS

Asynchrone Drosselung ist auf asynchrone Workflows zurückzuführen, bei denen Amazon ECS oder AWS CloudFormation möglicherweise in Ihrem Namen APIs aufrufen, um Ressourcen bereitzustellen. Es ist wichtig zu wissen, welche AWS APIs Amazon ECS in Ihrem Namen aufruft.

Die `CreateNetworkInterface` API wird beispielsweise für Aufgaben aufgerufen, die den `awsvpc` Netzwerkmodus verwenden, und die `DescribeTargetHealth` API wird aufgerufen, wenn Zustandsprüfungen für Aufgaben durchgeführt werden, die bei einem Load Balancer registriert sind.

Wenn Ihre Workloads ein beträchtliches Ausmaß erreichen, werden diese API-Operationen möglicherweise gedrosselt. Das heißt, sie werden möglicherweise so stark gedrosselt, dass sie die von Amazon ECS oder dem AWS-Service aufgerufenen System durchgesetzten Grenzwerte überschreiten. Wenn Sie beispielsweise Hunderte von Services bereitstellen, die jeweils Hunderte von Aufgaben gleichzeitig ausführen und den `awsvpc` Netzwerkmodus verwenden, ruft Amazon ECS Amazon EC2 EC2-API-Operationen wie `CreateNetworkInterface` und Elastic Load Balancing Balancing-API-Operationen wie `RegisterTarget` oder auf, `DescribeTargetHealth` um die elastic network interface bzw. den Load Balancer zu registrieren. Diese API-Aufrufe können die API-Grenzwerte überschreiten, was zu Drosselungsfehlern führen kann. Im Folgenden finden Sie ein Beispiel für einen Elastic Load Balancing Balancing-Drosselungsfehler, der in der Service-Event-Meldung enthalten ist.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Wenn diese API-Aufrufe dieselben Grenzwerte wie anderer API-Traffic in Ihrem Konto haben, kann es schwierig sein, sie zu überwachen, obwohl sie als Service-Ereignisse ausgegeben werden.

Drosselung des Monitors

Es ist wichtig zu ermitteln, welche API-Anfragen gedrosselt werden und wer diese Anfragen stellt. Sie können festlegen AWS CloudTrail , welche Monitore die Drosselung und Integration mit CloudWatch Amazon Athena und Amazon überwachen. EventBridge Sie können so konfigurieren CloudTrail ,

dass bestimmte Ereignisse an Logs gesendet werden. CloudWatch CloudWatch Logs, Log Insights, analysiert und analysiert die Ereignisse. Dadurch werden Details bei Drosselungsereignissen identifiziert, z. B. der Benutzer oder die IAM-Rolle, die den Anruf getätigt hat, und die Anzahl der getätigten API-Aufrufe. Weitere Informationen finden Sie unter Überwachen von CloudTrail Protokolldateien mit Protokollen. CloudWatch

Weitere Informationen zu CloudWatch Logs Insights und Anweisungen zum Abfragen von Protokolldateien finden Sie unter Logdaten mit CloudWatch Logs Insights analysieren.

Mit Amazon Athena können Sie Abfragen erstellen und Daten mit Standard-SQL analysieren. Sie können beispielsweise eine Athena-Tabelle erstellen, um Ereignisse zu analysieren CloudTrail . Weitere Informationen finden Sie unter Verwenden der CloudTrail Konsole zum Erstellen einer Athena-Tabelle für CloudTrail Protokolle.

Nachdem Sie eine Athena-Tabelle erstellt haben, können Sie einfache SQL-Abfragen wie die folgende verwenden, um `ThrottlingException` Fehler zu untersuchen.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail-table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2022-01-14T03:00:08Z' and '2022-01-23T07:15:08Z'
group by errorcode, awsregion, eventsource, username, eventname
order by count desc;
```

Amazon ECS sendet auch Ereignisbenachrichtigungen an Amazon EventBridge. Es gibt Ereignisse zur Änderung des Ressourcenstatus und Ereignisse bei Serviceaktionen. Dazu gehören API-Drosselungsereignisse wie und`ECS_OPERATION_THROTTLED`. `SERVICE_DISCOVERY_OPERATION_THROTTLED` Weitere Informationen finden Sie unter Aktionsereignisse des Amazon ECS-Service.

Diese Ereignisse können von einem Dienst genutzt werden, um beispielsweise als AWS Lambda Reaktion darauf Aktionen auszuführen. Weitere Informationen finden Sie unter Umgang mit Amazon ECS-Ereignissen.

Wenn Sie eigenständige Aufgaben ausführen, `RunTask` sind einige API-Operationen, z. B. asynchron, und Wiederholungsvorgänge werden nicht automatisch ausgeführt. In solchen Fällen können Sie Dienste wie AWS Step Functions die EventBridge Integration verwenden, um gedrosselte oder fehlgeschlagene Operationen erneut zu versuchen. Weitere Informationen finden Sie unter Container-Aufgaben verwalten (Amazon ECS, Amazon SNS).

Wird zur Überwachung CloudWatch der Drosselung verwendet

CloudWatch bietet die Überwachung der API-Nutzung für den **`Usage`** Namespace unter Nach Ressource. AWS Diese Metriken werden mit dem Typ API und dem Metriknamen CallCountprotokolliert. Sie können Alarme erstellen, die immer dann ausgelöst werden, wenn diese Metriken einen bestimmten Schwellenwert erreichen. Weitere Informationen finden Sie unter Visualisieren Ihrer Servicequotas und Einstellen von Alarmen.

CloudWatch bietet auch eine Erkennung von Anomalien. Diese Funktion verwendet maschinelles Lernen, um Basiswerte zu analysieren und festzulegen, die auf dem spezifischen Verhalten der Metrik basieren, für die Sie sie aktiviert haben. Bei ungewöhnlichen API-Aktivitäten können Sie diese Funktion zusammen mit CloudWatch Alarmen verwenden. Weitere Informationen finden Sie unter Verwenden der CloudWatch Anomalieerkennung.

Durch die proaktive Überwachung von Drosselungsfehlern können Sie sich an uns wenden, AWS Support um die entsprechenden Drosselungsgrenzwerte zu erhöhen und Unterstützung für Ihre individuellen Anwendungsanforderungen zu erhalten.

Gründe für den Ausfall der Amazon ECS-API

Wenn eine API-Aktion, die Sie über die Amazon ECS-API, -Konsole oder die ausgelöst haben, mit einer `failures` Fehlermeldung AWS CLI beendet wird, kann Folgendes bei der Behebung der Ursache hilfreich sein. Für den Fehler werden ein Grund und der Amazon-Ressourcenname (ARN) der Ressource zurückgegeben, die mit dem Fehler verknüpft ist.

Viele Ressourcen sind regionalspezifisch. Achten Sie also darauf, dass die Konsole auf die richtige Region für Ihre Ressourcen gesetzt ist. Stellen Sie bei der AWS CLI Verwendung von sicher, dass Ihre AWS CLI Befehle mit dem `--region` *`region`* Parameter an die richtige Region gesendet werden.

Weitere Informationen über die Struktur des `Failure`-Datentyps finden Sie unter Fehler in der Amazon Elastic Container Service-API-Referenz.

Im Folgenden finden Sie Beispiele für Fehlermeldungen, die Sie möglicherweise beim Ausführen von API-Befehlen erhalten.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
`DescribeClusters`	`MISSING`	Der angegebene Cluster wurde nicht gefunden. Überprüfen Sie die Schreibweise des Clusternamens.
`DescribeInstances`	`MISSING`	Die angegebene Container-Instance wurde nicht gefunden. Stellen Sie sicher, dass Sie den Cluster angegeben haben, in dem die Container-Instance registriert ist, und dass sowohl die Container-Instance-ARN als auch die ID korrekt sind.
`DescribeServices`	`MISSING`	Der angegebene Service wurde nicht gefunden. Stellen Sie sicher, dass der richtige Cluster oder die richtige Region angegeben ist und der Dienst-ARN oder -name gültig ist.
`DescribeTasks`	`MISSING`	Die angegebene Aufgabe wurde nicht gefunden. Überprüfen Sie, ob der richtige Cluster oder die richtige Region angegeben ist und ob sowohl der ARN oder die ID der Aufgabe gültig ist.
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	Bei `RESOURCE:CPU` -Fehlern ist die Anzahl der CPUs, die von der Aufgabe angefragt

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
		wurden, auf Ihrer gegebenen Container-Instance nicht verfügbar. Dies ist in der Regel der Fall, wenn die CPU-Einheitenanforderung in Ihrer Aufgabendefinition größer ist als die CPU-Größe der Amazon EC2 EC2-Instances, die in der Auto Scaling Scaling-Gruppe definiert sind, die dem Kapazitätsanbieter zugeordnet ist. Sie müssen die Konfiguration Ihres Kapazitätsanbieters überprüfen. Bei `RESOURCE:MEMORY` - Fehlern ist die Speicherm enge, die von der Aufgabe angefragt wurde, auf Ihrer gegebenen Container-Instance nicht verfügbar. Dies ist in der Regel der Fall, wenn der Speicherbedarf in Ihrer Aufgabendefinition größer ist als der unterstützte Speicher auf den Amazon EC2 EC2-Instances, die in der Auto Scaling Scaling-Gruppe definiert sind, die dem Kapazitätsanbieter zugeordnet ist. Sie müssen die Konfiguration Ihres Kapazitätsanbieters überprüfen.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
	`TaskFailedToStart: AGENT`	Die Container-Instance, auf der Sie eine Aufgabe zu starten versucht haben, hat einen Agenten, der derzeit nicht verbunden ist. Um langen Wartezeiten für die Platzierung der Aufgabe vorzubeugen, wurde die Anfrage zurückgewiesen. Informationen zur Fehlerbeh ebung finden Sie unter Wie behebe ich Probleme mit einem getrennten Amazon-EC S-Agenten.
	`TaskFailedToStart: MemberOf placement constraint unsatisfi ed`	Es gibt keine Container-Instance, die die in Ihrer Aufgabendefinition definierten Platzierungsbeschränkungen erfüllt.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
	`TaskFailedToStart: ATTRIBUTE`	Ihre Aufgabendefinition enthält einen Parameter, der ein spezifisches Container -Instance-Attribut erfordert , das auf Ihren Container-Instances nicht verfügbar ist. Beispiel: Wenn Ihre Aufgabe den Netzwerkmodus `awsvpc` verwendet, es aber keine Instances in Ihren angegeben en Subnetze mit dem Attribut `ecs.capability.task-eni` gibt. Weitere Informati onen darüber, welche Attribute für bestimmte Aufgabend efinitionsparameter und Agentenkonfigurationsvariab len erforderlich sind, finden Sie unter Amazon ECS-Aufga bendefinitionsparameter und Konfiguration des Amazon-EC S-Container-Agenten.
	`TaskFailedToStart: NO ACTIVE INSTANCES`	In Ihrem Kapazitätsanbieter gibt es keine aktiven Instances . Informationen über die Verwaltung Ihrer Auto-Scal ing-Gruppen finden Sie unter Auto-Scaling-Grupp en im Benutzerhandbuch für Amazon EC2 Auto Scaling.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
	`TaskFailedToStart: EMPTY CAPACITY PROVIDER`	Es gibt keine Instances in Ihrem Cluster. Dies liegt höchstwahrscheinlich an einem leeren Kapazität sanbieter oder daran, dass die Instances im Kapazität sanbieter nicht im Cluster registriert sind. Informationen über die Verwaltung Ihrer Auto-Scaling-Gruppen finden Sie unter Auto-Scaling-Gruppen im Benutzerhandbuch für Amazon EC2 Auto Scaling.
`GetTaskProtection`	`MISSING`	Die angegebene Aufgabe wurde nicht gefunden. Stellen Sie sicher, dass der Cluster-Name oder ARN und der Aufgaben-ARN sowie die ID gültig sind.
	`TASK_NOT_VALID`	Die angegebene Aufgabe ist nicht Teil eines Amazon ECS-Service. Nur vom Amazon-ECS-Service verwaltete Aufgaben können geschützt werden. Überprüfen Sie den Aufgaben-ARN oder die -ID und versuchen Sie es erneut.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
`RunTask` oder `StartTask`	`RESOURCE:*`	Die Ressource oder Ressourcen, die von der Aufgabe angefragt wurden, sind auf der gegebenen Container-Instance im Cluster nicht verfügbar. Wenn es sich bei der Ressource um CPU, Speicher, Ports oder Elastic-Network-Schnittstellen handelt, müssen Sie Ihrem Cluster möglicherweise Container-Instances hinzufügen. Für `RESOURCE:ENI` - Fehler stehen Ihrem Cluster keine verfügbaren Befestigungspunkte für Elastic-Network-Schnittstellen zur Verfügung, die für Aufgaben benötigt werden, die den Netzwerkmodus `awsvpc` verwenden. Amazon-EC2-Instances haben eine Begrenzung für die Anzahl der Netzwerkschnittstellen, die an sie angehängt werden können, und die primäre Netzwerkschnittstelle zählt als eine Einheit. Weitere Informationen darüber, wie viele Netzwerkschnittstellen für jeden Instance-Typ unterstützt werden, finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
		Instance-Typ im Amazon EC2 EC2-Benutzerhandbuch. Für `RESOURCE:GPU` -Fehler ist die Anzahl der GPUs, die von der Aufgabe angeforde rt werden, nicht verfügbar, und Sie müssen möglicher weise GPU-fähige Container -Instances zu Ihrem Cluster hinzufügen. Weitere Informati onen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.
	`AGENT`	Die Container-Instance, auf der Sie eine Aufgabe zu starten versucht haben, hat einen Agenten, der derzeit nicht verbunden ist. Um langen Wartezeiten für die Platzierung der Aufgabe vorzubeugen, wurde die Anfrage zurückgewiesen. Informationen zur Fehlerbeh ebung finden Sie unter Wie behebe ich Probleme mit einem getrennten Amazon-EC S-Agenten.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
	`LOCATION`	Die Container-Instance, auf der Sie versucht haben, eine Aufgabe zu starten, befindet sich in einer anderen Availability Zone als die Subnetze, die Sie in `awsVpcConfiguration` angegeben haben.
	`ATTRIBUTE`	Ihre Aufgabendefinition enthält einen Parameter, der ein spezifisches Container-Instance-Attribut erfordert, das auf Ihren Container-Instances nicht verfügbar ist. Beispiel: Wenn Ihre Aufgabe den Netzwerkmodus `awsvpc` verwendet, es aber keine Instances in Ihren angegebenen Subnetze mit dem Attribut `ecs.capability.task-eni` gibt. Weitere Informationen darüber, welche Attribute für bestimmte Aufgabendefinitionsparameter und Agentenkonfigurationsvariablen erforderlich sind, finden Sie unter Amazon ECS-Aufgabendefinitionsparameter und Konfiguration des Amazon-ECS-Container-Agenten.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
`StartTask`	`MISSING`	Die Container-Instance, auf der Sie versucht haben, die Aufgabe zu starten, wurde nicht gefunden. Prüfen Sie, ob der falsche Cluster oder die falsche Region angegeben ist oder ob der ARN oder die ID der Container-Instance falsch geschrieben ist.
	`INACTIVE`	Die Container-Instance, auf der Sie versucht haben, die Aufgabe zu starten, wurde vorher von Amazon ECS abgemeldet und kann nicht verwendet werden.
`UpdateTaskProtection`	`DEPLOYMENT_BLOCKED`	Der Task-Schutz kann nicht eingerichtet werden, da eine oder mehrere geschützt e Aufgaben verhindern, dass die Dienstbereitstellu ng einen stabilen Status erreicht. Deaktivieren Sie den Aufgabenschutz für vorhanden e Aufgaben oder warten Sie, bis der Aufgabenschutz abläuft.

API-Aktion	Grund des Ausfalls oder Grund des Anhaltens	Ursache
	`MISSING`	Die angegebene Aufgabe wurde nicht gefunden. Stellen Sie sicher, dass der Cluster-Name oder ARN und der Aufgaben-ARN sowie die ID gültig sind.
	`TASK_NOT_VALID`	Die angegebene Aufgabe ist nicht Teil eines Amazon ECS-Service. Nur vom Amazon-ECS-Service verwaltete Aufgaben können geschützt werden. Überprüfen Sie den Aufgaben-ARN oder die -ID und versuchen Sie es erneut.

> **Note**
>
> Neben den hier beschriebenen Fehlerszenarien können API-Operationen auch aufgrund von Ausnahmen fehlschlagen, was zu Fehlerreaktionen führt. Eine Liste solcher Ausnahmen finden Sie unter Häufige Fehler.

Sicherheit im Amazon Elastic Container Service

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS -Compliance-Programme regelmäßig. Weitere Informationen zu den für Amazon Elastic Container Service geltenden Compliance-Programmen finden Sie unter Im Rahmen des Compliance-Programms zugelassene AWS -Services.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation zeigt Ihnen, wie Sie das Modell der übergreifenden Verantwortlichkeit bei der Verwendung von Amazon ECS einsetzen können. Die folgenden Themen veranschaulichen, wie Sie Amazon ECS zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie lernen auch, wie Sie andere AWS Dienste nutzen können, die Sie bei der Überwachung und Sicherung Ihrer Amazon ECS-Ressourcen unterstützen.

Themen

Identitäts- und Zugriffsverwaltung für Amazon Elastic Container Service

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren steuern, wer authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) ist, Amazon-ECS-Ressourcen zu nutzen. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in Amazon ECS ausführen.

Service-Benutzer – Wenn Sie den Amazon-ECS-Service zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen bereit, die Sie benötigen. Wenn Sie zur Ausführung von Aufgaben weitere Amazon-ECS-Features verwenden, benötigen

Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Wenn Sie auf ein Feature in Amazon ECS nicht zugreifen können, siehe Identitäts- und Zugriffsfehler bei Amazon Elastic Container Service.

Service-Administrator – Wenn Sie in Ihrem Unternehmen für die Amazon-ECS-Ressourcen zuständig sind, haben Sie wahrscheinlich vollen Zugriff auf Amazon ECS. Ihre Aufgabe besteht darin, zu bestimmen, auf welche Amazon-ECS-Features und -Ressourcen Ihre Service-Nutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit Amazon ECS verwenden kann, finden Sie unter So funktioniert Amazon Elastic Container Service mit IAM.

IAM-Administrator – Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Amazon ECS verfassen können. Beispiele für identitätsbasierte Amazon-ECS-Richtlinien, die Sie in IAM verwenden können, finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service.

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, mit denen Sie Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch signieren können. Wenn Sie keine AWS Tools verwenden,

müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu

IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert,

so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

- Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

- Auf Amazon EC2 ausgeführte Anwendungen — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen

auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service Control Policies (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und

der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter Bewertungslogik für Richtlinien.

So funktioniert Amazon Elastic Container Service mit IAM

Bevor Sie mit IAM den Zugriff auf Amazon ECS verwalten können, sollten Sie sich darüber informieren, welche IAM-Features Sie mit Amazon ECS verwenden können.

IAM-Features, die Sie mit Amazon Elastic Container Service verwenden können

IAM-Feature	Amazon-ECS-Support
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Teilweise
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Ja
Temporäre Anmeldeinformationen	Ja
Forward Access Sessions (FAS)	Ja
Servicerollen	Ja

IAM-Feature	Amazon-ECS-Support
Service-verknüpfte Rollen	Ja

Einen allgemeinen Überblick darüber, wie Amazon ECS und andere AWS Services mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Services, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für Amazon ECS

Unterstützt Richtlinien auf Identitätsbasis.	Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Amazon ECS

Beispiele für identitätsbasierte Amazon-ECS-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service.

Ressourcenbasierte Richtlinien in Amazon ECS

Unterstützt ressourcenbasierte Richtlinien	Nein

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und

Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM-Administrator des vertrauenswürdigen Kontos auch der Prinzipalentität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter Wie sich IAM-Rollen von ressourcenbasierten Richtlinien unterscheiden im IAM-Benutzerhandbuch.

Richtlinienaktionen für Amazon ECS

Unterstützt Richtlinienaktionen	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der Amazon-ECS-Aktionen finden Sie unter Von Amazon Elastic Container Service definierte Aktionen in der Service-Authorization-Referenz.

Richtlinienaktionen in Amazon ECS verwenden das folgende Präfix vor der Aktion:

```
ecs
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "ecs:action1",
      "ecs:action2"
         ]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Describe` beginnen, einschließlich der folgenden Aktion:

```
"Action": "ecs:Describe*"
```

Beispiele für identitätsbasierte Amazon-ECS-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service.

Richtlinienressourcen für Amazon ECS

Unterstützt Richtlinienressourcen	Teilweise

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Amazon-ECS-Ressourcentypen und ihrer ARNs finden Sie unter Von Amazon Elastic Container Service definierte Ressourcen in der Service-Authorization-Referenz. Sie erfahren, mit welchen Aktionen Sie den ARN einer jeden Ressource angeben können, unter Von Amazon Elastic Container Service definierte Aktionen.

Einige API-Aktionen von Amazon ECS unterstützen mehrere Ressourcen. Beispiel: Beim Aufruf der API-Aktion `DescribeClusters` kann auf mehrere Cluster verwiesen werden. Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, trennen Sie die ARNs durch Kommata voneinander.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Die Amazon-ECS-Cluster-Ressource verfügt zum Beispiel über den folgenden ARN:

```
arn:${Partition}:ecs:${Region}:${Account}:cluster/${clusterName}
```

Verwenden Sie die folgenden ARNs, um `my-cluster-1`- und `my-cluster-2`-Cluster in Ihrer Anweisung anzugeben:

```
"Resource": [
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-1",
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-2"
```

Um alle Cluster anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*):

```
"Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/*"
```

Für Aufgabendefinitionen können Sie die neueste Revision oder eine bestimmte Revision angeben.

Um alle Versionen der Aufgabendefinition anzugeben, verwenden Sie den Platzhalter (*):

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:*"
```

Um eine bestimmte Version der Aufgabendefinition anzugeben, verwenden Sie $ {}: TaskDefinition RevisionNumber

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:${TaskDefinitionRevisionNumber}"
```

Beispiele für identitätsbasierte Amazon-ECS-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service.

Richtlinien-Bedingungsschlüssel für Amazon ECS

Unterstützt servicespezifische Richtlini enbedingungsschlüssel	Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die Bedingung mithilfe einer logischen `OR` Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Amazon ECS unterstützt die folgenden servicespezifischen Bedingungsschlüssel, die Sie verwenden können, um eine feinabgestimmte Filterung für Ihre IAM-Richtlinien bereitzustellen:

Bedingungsschlüssel	Beschreibung	Bewertungstypen
aws: RequestTag /$ {} TagKey	Dieser Kontextschlüssel liegt im Format `"aws:RequestTag/ tag-key":"tag-value "` vor. Dabei bezeichnen *`tag-key`* und *`tag-value`* ein Paar aus Tag-Schlüssel und -Wert. Überprüft, ob das Tag-Schlüssel-Wert-Paar in einer Anfrage vorhanden ist. AWS Sie können z. B. prüfen, ob die Anforderung den Tag-Schlüssel `"Dept"` enthält und dieser den Wert `"Accounting"` hat.	String
aws: ResourceTag /$ {} TagKey	Dieser Kontextschlüssel liegt im Format `"aws:ResourceTag/ tag-key":"tag-value "` vor. Dabei bezeichnen *`tag-key`* und *`tag-value`* ein Paar aus Tag-Schlüssel und -Wert. Überprüft, ob das an die Identitätsressource (Benutzer oder Rolle) angefügte Tag mit dem angegebenen Schlüsselnamen und Wert übereinstimmt.	String
war: TagKeys	Dieser Kontextschlüssel liegt im Format `"aws:TagKeys":" tag-key"` vor. Dabei ist *`tag-key`* eine Liste von Tag-Schlüsseln ohne Werte (z. B. `["Dept","Cost-Center"]`). Prüft die Tag-Schlüssel, die in einer AWS Anfrage vorhanden sind.	String
ecs: ResourceTag /$ {} TagKey	Dieser Kontextschlüssel liegt im Format `"ecs:ResourceTag/ tag-key":"tag-value "` vor. Dabei	String

Bedingungsschlüssel	Beschreibung	Bewertungstypen
	bezeichnen *tag-key* und *tag-value* ein Paar aus Tag-Schlüssel und -Wert. Überprüft, ob das an die Identitätsressource (Benutzer oder Rolle) angefügte Tag mit dem angegebenen Schlüsselnamen und Wert übereinstimmt.	
ecs:cluster	Der Kontextschlüssel liegt im Format `"ecs:cluster":"` *cluster-arn* `"` vor, wobei *cluster-arn* der ARN für den Amazon-ECS-Cluster ist.	ARN, Null
ecs:container-instances	Der Kontextschlüssel liegt im Format `"ecs:container-instances":"` *container-instance-arns*`"` vor, wobei dem *container-instance-arns* ein oder mehrere Container-Instance-ARNs ist/sind.	ARN, Null
ecs:container-name	Der Kontextschlüssel hat das Format `"ecs:container-name":"` *container-name* `"`, wobei *container-instance-* der Name eines Amazon-ECS-Containers ist, der in der Aufgabendefinition definiert ist.	String
ecs:enable-execute-command	Der Kontextschlüssel hat das Format `"ecs:enable-execute-command":"` *value*`"`, wobei *value-* „`true`“ oder „`false`“ ist.	String
ecs:enable-service-connect	Der Kontextschlüssel hat das Format `"ecs:enable-service-connect":"` *value*`"`, wobei *value* `"true"` oder `"false"` ist.	String
ecs:ebs-volumes aktivieren	Der Kontextschlüssel hat das Format `"ecs:enable-ebs-volumes":"` *value*`"`, wobei *value* `"true"` oder `"false"` ist.	String

Bedingungsschlüssel	Beschreibung	Bewertungstypen
ecs:namespace	Der Kontextschlüssel hat das Format `"ecs:namespace":" namespace-arn "`, wobei *namespace-arn* der ARN für den AWS Cloud Map -Namespace ist.	ARN, Null
ecs:service	Der Kontextschlüssel liegt im Format `"ecs:service":" service-arn "` vor, wobei *service-arn* der ARN für den Amazon-ECS-Service ist.	ARN, Null
ecs:task-definition	Der Kontextschlüssel liegt im Format `"ecs:task-definition":" task-definition-arn "` vor, wobei *task-definition-arn* der ARN für die Amazon-ECS-Aufgabendefinition ist.	ARN, Null
ecs:account-setting	Der Kontextschlüssel ist als `"ecs:account-setting":" account-setting "` formatiert, wobei *Kontoeinstellung* der Name einer Amazon-ECS-Kontoeinstellung ist.	String

Eine Liste der Amazon-ECS-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Elastic Container Service in der Service-Authorization-Referenz. Um zu erfahren, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, siehe Von Amazon Elastic Container Service definierte Aktionen.

Beispiele für identitätsbasierte Amazon-ECS-Richtlinien finden Sie unter Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service.

Zugriffssteuerungslisten (ACLs) in Amazon ECS

Unterstützt ACLs	Nein

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Attributbasierte Zugriffssteuerung (ABAC) mit Amazon ECS

> **Important**
>
> Amazon ECS unterstützt die attributbasierte Zugriffskontrolle für alle Amazon ECS-Ressourcen. Um festzustellen, ob Sie Attribute für die Reichweite einer Aktion verwenden können, verwenden Sie die Tabelle Von Amazon ECS definierte Aktionen in der Service-Autorisierungsreferenz. Stellen Sie zunächst sicher, dass sich in der Spalte Ressource eine Ressource befindet. Verwenden Sie dann die Spalte Bedingungsschlüssel, um die Schlüssel für die Kombination aus Aktion und Ressource anzuzeigen.

Unterstützt ABAC (Tags in Richtlinien)	Ja

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Was ist ABAC? im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Weitere Informationen über das Markieren von Amazon-ECS-Ressourcen mit Tags finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter Beschreiben von Amazon-ECS-Services auf der Grundlage von Tags.

Verwenden temporärer Anmeldeinformationen mit Amazon ECS

Unterstützt temporäre Anmeldeinformationen	Ja

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services funktionieren AWS-Services , finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter Wechseln zu einer Rolle (Konsole) im IAM-Benutzerhandbuch.

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen in IAM.

Zugriffssitzungen für Amazon ECS weiterleiten

Unterstützt Forward Access Sessions (FAS)	Ja

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-

Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für Amazon ECS

Unterstützt Servicerollen	Ja

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle könnte die Amazon-ECS-Funktionalität beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Amazon ECS dazu Anleitungen gibt.

Serviceverknüpfte Rollen für Amazon ECS

Unterstützt serviceverknüpfte Rollen	Ja

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Amazon-ECS-Rollen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.

Beispiele für identitätsbasierte Richtlinien für Amazon Elastic Container Service

Benutzer besitzen standardmäßig keine Berechtigungen zum Erstellen oder Ändern von Amazon-ECS-Ressourcen. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Einzelheiten zu Aktionen und Ressourcentypen, die von Amazon ECS definiert werden, einschließlich des Formats der ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Elastic Container Service in der Service-Authorisierungsreferenz.

Themen

Bewährte Methoden für Amazon ECS-Richtlinien

Identitätsbasierte Richtlinien können festlegen, ob jemand Amazon-ECS-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder daraus löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.

- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Erlauben Sie Amazon ECS-Benutzern, ihre eigenen Berechtigungen einzusehen

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
```

```
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Beispiele für Amazon ECS-Cluster

Die folgende IAM-Richtlinie erteilt die Berechtigung, Cluster zu erstellen und aufzulisten. Die Aktionen `CreateCluster` und `ListClusters` akzeptieren keine Ressourcen, daher wird die Ressourcendefinition auf * für alle Ressourcen festgelegt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:ListClusters"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Die folgende IAM-Richtlinie erteilt die Berechtigung, ein bestimmtes Cluster zu beschreiben und zu löschen. Die Aktionen `DescribeClusters` und `DeleteCluster` akzeptieren Cluster-ARNs als Ressourcen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DescribeClusters",
                "ecs:DeleteCluster"
            ],
```

```
            "Resource": ["arn:aws:ecs:us-east-1:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

Die folgende IAM-Richtlinie kann einem Benutzer oder einer Gruppe angefügt werden und sie erlaubt nur diesem Benutzer oder dieser Gruppe, Operationen auf einem bestimmten Cluster auszuführen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:<aws_account_id>:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:<aws_account_id>:cluster/default"}
            }
        }
    ]
}
```

Beispiele für Amazon ECS-Container-Instances

Die Registrierung von Container-Instances erfolgt durch den Amazon-ECS-Agenten, aber es kann vorkommen, dass Sie einem Benutzer erlauben möchten, eine Instance manuell von einem Cluster abzumelden. Vielleicht war die Container-Instance zufällig beim falschen Cluster registriert oder die Instance wurde beendet, obwohl noch Aufgaben auf ihr ausgeführt wurden.

Die folgenden IAM-Richtlinie erlaubt es dem Benutzer, Container-Instances in einem bestimmten Cluster aufzulisten und deren Registrierung aufzuheben:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances"
            ],
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

Die folgende IAM-Richtlinie erlaubt es einem Benutzer, eine angegebene Container-Instance in einem angegebenen Cluster zu beschreiben. Um diese Berechtigung für alle Container-Instances in einem Cluster zu öffnen, können Sie die Container-Instance-UUID durch * ersetzen.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeContainerInstances"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/
<cluster_name>/<container_instance_UUID>"]
        }
    ]
}
```

Beispiele für Amazon ECS-Aufgabendefinitionen

IAM-Richtlinien zur Aufgabendefinition unterstützen Berechtigungen auf Ressourcenebene nicht, aber die folgende IAM-Richtlinie erlaubt es einem Benutzer, Aufgabendefinitionen zu registrieren, aufzulisten und zu beschreiben:

Wenn Sie die Konsole verwenden, müssen Sie `CloudFormation: CreateStack` als `Action` hinzufügen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RegisterTaskDefinition",
                "ecs:ListTaskDefinitions",
                "ecs:DescribeTaskDefinition"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Beispiel für eine Amazon ECS-Aufgabe ausführen

Die Ressourcen für `RunTask` sind Aufgabendefinitionen. Um einzuschränken, auf welchen Clustern ein Benutzer Aufgabendefinitionen ausführen kann, können Sie diese in dem Block `Condition` angeben. Der Vorteil ist, dass Sie weder Aufgabendefinitionen noch Cluster in Ihren Ressourcen

auflisten müssen, um den entsprechenden Zugriff zu ermöglichen. Sie können eines, das andere oder beides anwenden.

Die folgende IAM-Richtlinie erteilt die Berechtigung, jede Revision einer bestimmten Aufgabendefinition auf einem bestimmten Cluster auszuführen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Beispiel für eine Amazon ECS-Aufgabe starten

Die Ressourcen für `StartTask` sind Aufgabendefinitionen. Um einzuschränken, auf welchen Clustern und Container-Instances ein Benutzer Aufgabendefinitionen starten kann, können Sie diese in dem Block `Condition` angeben. Der Vorteil ist, dass Sie weder Aufgabendefinitionen noch Cluster in Ihren Ressourcen auflisten müssen, um den entsprechenden Zugriff zu ermöglichen. Sie können eines, das andere oder beides anwenden.

Die folgende IAM-Richtlinie erteilt die Berechtigung, jede Revision einer bestimmten Aufgabendefinition auf einem bestimmten Cluster und einer bestimmten Container-Instance zu starten.

> **Note**
>
> Wenn Sie in diesem Beispiel die `StartTask` API mit dem AWS CLI oder einem anderen AWS SDK aufrufen, müssen Sie die Revision der Aufgabendefinition angeben, damit die `Resource` Zuordnung übereinstimmt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:StartTask"],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:container-instances":
 ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/<cluster_name>/
<container_instance_UUID>"]
                }
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Auflisten und Beschreiben von Amazon ECS-Aufgabenbeispielen

Die folgende IAM-Richtlinie erlaubt es einem Benutzer, Aufgaben für ein angegebenes Cluster aufzulisten:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:ListTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

Die folgende IAM-Richtlinie erlaubt es einem Benutzer, eine angegebene Aufgabe in einem angegebenen Cluster zu beschreiben:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task/<cluster_name>/
<task_UUID>"]
        }
    ]
}
```

Beispiel für einen Amazon ECS-Service erstellen

Die folgende IAM-Richtlinie erlaubt es einem Benutzer, Amazon-ECS-Services in AWS Management Console zu erstellen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:CreateService",
                "elasticloadbalancing:Describe*",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
```

```
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Beispiel für einen Amazon ECS-Service aktualisieren

Die folgende IAM-Richtlinie erlaubt es einem Benutzer, Amazon-ECS-Services in der AWS Management Console zu aktualisieren:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:UpdateService",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Beschreiben von Amazon-ECS-Services auf der Grundlage von Tags

Sie können Bedingungen in Ihrer identitätsbasierten Richtlinie verwenden, um den Zugriff auf Amazon-ECS-Ressourcen basierend auf Tags zu steuern. In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die die Beschreibung Ihrer Services ermöglicht. Die Berechtigung wird jedoch nur gewährt, wenn der Wert des `Owner`-Tags der Name des Benutzers ist. Diese Richtlinie gewährt auch die Berechtigungen, die für die Ausführung dieser Aktion auf der Konsole erforderlich sind.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeServices",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "*"
        },
        {
            "Sid": "ViewServiceIfOwner",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "arn:aws:ecs:*:*:service/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Sie können diese Richtlinie den IAM-Benutzern in Ihrem Konto anfügen. Wenn ein Benutzer mit dem Namen `richard-roe` versucht, einen Amazon-ECS-Service zu beschreiben, muss der Service mit `Owner=richard-roe` oder `owner=richard-roe` markiert sein. Andernfalls wird der Zugriff abgelehnt. Der Tag-Schlüssel `Owner` der Bedingung stimmt sowohl mit `Owner` als auch mit `owner` überein, da die Namen von Bedingungsschlüsseln nicht zwischen Groß- und Kleinschreibung unterscheiden. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

Beispiel zum Überschreiben des Amazon ECS Service Connect-Namespace verweigern

Die folgende IAM-Richtlinie verweigert einem Benutzer das Überschreiben des standardmäßigen Service-Connect-Namespace in einer Service-Konfiguration. Der Standard-Namespace ist im Cluster festgelegt. Sie können es jedoch in einer Service-Konfiguration überschreiben. Aus Konsistenzgründen sollten Sie erwägen, all Ihre neuen Services so einzustellen, dass sie denselben Namespace verwenden. Verwenden Sie die folgenden Kontextschlüssel, um Services zur Verwendung eines bestimmten Namespace zu verpflichten. Ersetzen Sie im folgenden Beispiel `<region>`, `<aws_account_id>`, `<cluster_name>` und `<namespace_id>` mit Ihren eigenen Werten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:namespace":
 "arn:aws:servicediscovery:<region>:<aws_account_id>:namespace/<namespace_id>"
                }
            },
            "Resource": "*"
        }
    ]
}
```

AWS verwaltete Richtlinien für Amazon Elastic Container Service

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, AWS verwaltete Richtlinien zu verwenden, als Richtlinien selbst zu schreiben. Es erfordert Zeit und Fachwissen,

um von Kunden verwaltete IAM-Richtlinien zu erstellen, die Ihrem Team nur die benötigten Berechtigungen bieten. Um schnell loszulegen, können Sie unsere AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem AWS Konto verfügbar. Weitere Informationen zu AWS verwalteten Richtlinien finden Sie unter AWS Verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS Dienste verwalten und aktualisieren AWS verwaltete Richtlinien. Sie können die Berechtigungen in AWS verwalteten Richtlinien nicht ändern. Dienste fügen einer AWS verwalteten Richtlinie gelegentlich zusätzliche Berechtigungen hinzu, um neue Funktionen zu unterstützen. Diese Art von Update betrifft alle Identitäten (Benutzer, Gruppen und Rollen), an welche die Richtlinie angehängt ist. Es ist sehr wahrscheinlich, dass Dienste eine AWS verwaltete Richtlinie aktualisieren, wenn eine neue Funktion eingeführt wird oder wenn neue Operationen verfügbar werden. Dienste entfernen keine Berechtigungen aus einer AWS verwalteten Richtlinie, sodass durch Richtlinienaktualisierungen Ihre bestehenden Berechtigungen nicht beeinträchtigt werden.

AWS Unterstützt außerdem verwaltete Richtlinien für Jobfunktionen, die sich über mehrere Dienste erstrecken. Die AWS verwaltete ReadOnlyAccess-Richtlinie bietet beispielsweise Lesezugriff auf alle AWS Dienste und Ressourcen. Wenn ein Dienst ein neues Feature startet, werden nur Leseberechtigungen für neue Operationen und Ressourcen AWS hinzugefügt. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in Verwaltete AWS -Richtlinien für Auftragsfunktionen im IAM-Leitfaden.

Amazon ECS und Amazon ECR bieten mehrere verwaltete Richtlinien und Vertrauensstellungen, die Sie Benutzern, Gruppen, Rollen oder Amazon-EC2-Instances und Amazon-ECS-Aufgaben anfügen können, um verschiedene Kontrollstufen für Ressourcen und API-Operationen festzulegen. Sie können diese Richtlinien direkt anwenden oder als Ausgangspunkt nutzen, um eigene Richtlinien zu erstellen. Weitere Informationen zu verwalteten Amazon ECR-Richtlinien finden Sie unter Verwaltete Amazon ECR-Richtlinien.

AmazonECS_ FullAccess

Sie können die `AmazonECS_FullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen.

Diese Richtlinie gewährt Administratorzugriff auf Amazon ECS-Ressourcen und gewährt einer IAM-Identität (z. B. einem Benutzer, einer Gruppe oder Rolle) Zugriff auf die AWS Services, in die Amazon ECS integriert ist, um alle Amazon ECS-Funktionen nutzen zu können. Die Verwendung dieser Richtlinie ermöglicht den Zugriff auf alle Amazon-ECS-Features, die in AWS Management Console verfügbar sind.

Details zu Berechtigungen

Die von `AmazonECS_FullAccess` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Nach den bewährten Methoden zur Erteilung von geringsten Privilegien können Sie die von `AmazonECS_FullAccess` verwaltete Richtlinie als Vorlage zum Erstellen eigener benutzerdefinierter Richtlinien verwenden. Auf diese Weise können Sie Berechtigungen zu und aus der verwalteten Richtlinie basierend auf Ihren spezifischen Anforderungen entfernen oder hinzufügen.

- `ecs`— Ermöglicht Principals vollen Zugriff auf alle Amazon ECS-API-Operationen.
- `application-autoscaling`: Ermöglicht es Prinzipalen, Application Auto Scaling Ressourcen zu erstellen, zu beschreiben und zu verwalten. Dies ist erforderlich, wenn Sie das Service-Auto-Scaling für Ihre Amazon-ECS-Dienste aktivieren.
- `appmesh`: Ermöglicht es Prinzipalen, App Mesh-Service-Netze und virtuelle Knoten aufzulisten und virtuelle App-Mesh-Knoten zu beschreiben. Dies ist erforderlich, wenn Sie Ihre Amazon-ECS-Dienste in App Mesh integrieren.
- `autoscaling`: Ermöglicht es Prinzipalen, Amazon EC2 Auto Scaling Ressourcen zu erstellen, zu verwalten und zu beschreiben. Dies ist erforderlich, wenn Sie Amazon EC2 Auto Scaling Scaling-Gruppen verwalten und die Cluster-Auto-Scaling-Funktion verwenden.
- `cloudformation`— Ermöglicht Prinzipalen das Erstellen und Verwalten AWS CloudFormation von Stacks. Dies ist erforderlich, wenn Sie Amazon-ECS-Cluster mit AWS Management Console erstellen und anschließend dieser Cluster verwalten.
- `cloudwatch`— Ermöglicht es Prinzipalen, CloudWatch Amazon-Alarme zu erstellen, zu verwalten und zu beschreiben.
- `codedeploy`— Ermöglicht Prinzipalen, Anwendungsbereitstellungen zu erstellen und zu verwalten und deren Konfigurationen, Versionen und Bereitstellungsziele einzusehen.
- `sns`: Ermöglicht es Prinzipalen, eine Liste mit Amazon SNS-Themen anzuzeigen.
- `lambda`: Ermöglicht es Prinzipalen, eine Liste von AWS Lambda -Funktionen und deren versionsspezifischen Konfigurationen anzuzeigen.
- `ec2`— Ermöglicht Principals, Amazon EC2 EC2-Instances auszuführen und Routen, Routing-Tabellen, Internet-Gateways, Startgruppen, Sicherheitsgruppen, virtuelle private Clouds, Spot-Flotten und Subnetze zu erstellen und zu verwalten.
- `elasticloadbalancing`: Ermöglicht Prinzipalen das Erstellen, Beschreiben und Löschen von Elastic Load Balancing-Lastenausgleichsdiensten. Prinzipale können auch Tags zu neu erstellten Zielgruppen, Listener und Listener-Regeln für Load Balancer hinzufügen.

- `events`— Ermöglicht Prinzipalen das Erstellen, Verwalten und Löschen von EventBridge Amazon-Regeln und deren Zielen.
- `iam`: Ermöglicht es Prinzipalen, IAM-Rollen und ihre angehängten Richtlinien aufzulisten. Prinzipale können auch Instance-Profile auflisten, die für Ihre Amazon-EC2-Instances verfügbar sind.
- `logs`— Ermöglicht Prinzipalen das Erstellen und Beschreiben von Amazon CloudWatch Logs-Protokollgruppen. Prinzipale können auch Protokollereignisse für diese Protokollgruppen auflisten.
- `route53`: Ermöglicht Prinzipalen das Erstellen, Verwalten und Löschen von gehosteten Amazon Route 53-Zonen. Prinzipale können auch die Konfiguration und Informationen zur Amazon Route 53 Integritätsprüfung anzeigen. Weitere Informationen über gehostete Zonen finden Sie unter Arbeiten mit gehosteten Zonen.
- `servicediscovery`— Ermöglicht Prinzipalen, AWS Cloud Map Dienste zu erstellen, zu verwalten und zu löschen und private DNS-Namespaces zu erstellen.

Es folgt eine Beispielrichtlinie `AmazonECS_FullAccess`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DescribeScalableTargets",
                "application-autoscaling:DescribeScalingActivities",
                "application-autoscaling:DescribeScalingPolicies",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "appmesh:DescribeVirtualGateway",
                "appmesh:DescribeVirtualNode",
                "appmesh:ListMeshes",
                "appmesh:ListVirtualGateways",
                "appmesh:ListVirtualNodes",
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:Describe*",
```

```
                "autoscaling:UpdateAutoScalingGroup",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:PutMetricAlarm",
                "codedeploy:BatchGetApplicationRevisions",
                "codedeploy:BatchGetApplications",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:ContinueDeployment",
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
                "codedeploy:GetApplicationRevision",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:StopDeployment",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CancelSpotFleetRequests",
                "ec2:CreateInternetGateway",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSubnet",
                "ec2:CreateVpc",
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteSubnet",
```

```
                "ec2:DeleteVpc",
                "ec2:Describe*",
                "ec2:DetachInternetGateway",
                "ec2:DisassociateRouteTable",
                "ec2:ModifySubnetAttribute",
                "ec2:ModifyVpcAttribute",
                "ec2:RequestSpotFleet",
                "ec2:RunInstances",
                "ecs:*",
                "elasticfilesystem:DescribeAccessPoints",
                "elasticfilesystem:DescribeFileSystems",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateRule",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:DeleteListener",
                "elasticloadbalancing:DeleteLoadBalancer",
                "elasticloadbalancing:DeleteRule",
                "elasticloadbalancing:DeleteTargetGroup",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:DescribeTargetGroups",
                "events:DeleteRule",
                "events:DescribeRule",
                "events:ListRuleNamesByTarget",
                "events:ListTargetsByRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "fsx:DescribeFileSystems",
                "iam:ListAttachedRolePolicies",
                "iam:ListInstanceProfiles",
                "iam:ListRoles",
                "lambda:ListFunctions",
                "logs:CreateLogGroup",
                "logs:DescribeLogGroups",
                "logs:FilterLogEvents",
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetHealthCheck",
                "route53:GetHostedZone",
                "route53:ListHostedZonesByName",
                "servicediscovery:CreatePrivateDnsNamespace",
```

```
                "servicediscovery:CreateService",
                "servicediscovery:DeleteService",
                "servicediscovery:GetNamespace",
                "servicediscovery:GetOperation",
                "servicediscovery:GetService",
                "servicediscovery:ListNamespaces",
                "servicediscovery:ListServices",
                "servicediscovery:UpdateService",
                "sns:ListTopics"
            ],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "ssm:GetParameters",
                "ssm:GetParametersByPath"
            ],
            "Resource": "arn:aws:ssm:*:*:parameter/aws/service/ecs*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteInternetGateway",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSecurityGroup"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"ec2:ResourceTag/aws:cloudformation:stack-name":
"EC2ContainerService-*"}
            }
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
```

```
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInstanceRole*"],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn"
                    ]
                }
            }
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsAutoscaleRole*"],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "application-autoscaling.amazonaws.com",
                        "application-autoscaling.amazonaws.com.cn"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": [
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com",
                        "ecs.application-autoscaling.amazonaws.com",
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": ["elasticloadbalancing:AddTags"],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "elasticloadbalancing:CreateAction": [
                        "CreateTargetGroup",
                        "CreateRule",
                        "CreateListener",
                        "CreateLoadBalancer"
                    ]
                }
            }
        }
    ]
}
```

AmazonECS ECS-Volumen InfrastructureRole PolicyFor

Die `AmazonECSInfrastructureRolePolicyForVolumes` verwaltete IAM-Richtlinie gewährt die Berechtigungen, die Amazon ECS benötigt, um AWS API-Aufrufe in Ihrem Namen durchzuführen. Sie können diese Richtlinie an die IAM-Rolle anhängen, die Sie mit Ihrer Volume-Konfiguration angeben, wenn Sie Amazon ECS-Aufgaben und -Services starten. Diese Rolle ermöglicht es Amazon ECS, Volumes zu verwalten, die mit Ihren Aufgaben verknüpft sind. Weitere Informationen finden Sie unter Amazon ECS-Infrastruktur-IAM-Rolle.

Details zu Berechtigungen

Die von `AmazonECSInfrastructureRolePolicyForVolumes` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Gemäß den standardmäßigen Sicherheitsempfehlungen zur Gewährung der geringsten Rechte können Sie die `AmazonECSInfrastructureRolePolicyForVolumes` verwaltete Richtlinie als Vorlage für die Erstellung Ihrer eigenen benutzerdefinierten Richtlinie verwenden, die nur die von Ihnen benötigten Berechtigungen enthält.

- `ec2:CreateVolume`— Ermöglicht es einem Principal, ein Amazon EBS-Volume zu erstellen, und zwar genau dann, wenn es mit den `AmazonECSManaged` Tags `AmazonECSCreated` und gekennzeichnet ist. Diese Berechtigung ist erforderlich, um Amazon EBS-Volumes zu erstellen, die Amazon ECS-Aufgaben zugeordnet sind, und um die Anzahl der Berechtigungen zu minimieren, die Amazon ECS durch diese Richtlinie gewährt werden.
- `ec2:CreateTags`— Ermöglicht einem Principal das Hinzufügen von Tags zu einem Amazon EBS-Volume als Teil von`ec2:CreateVolume`. Diese Genehmigung wird von Amazon ECS

benötigt, um kundenspezifische Tags zu Amazon EBS-Volumes hinzuzufügen, die in Ihrem Namen erstellt wurden.

- `ec2:AttachVolume`— Ermöglicht einem Principal, ein Amazon EBS-Volume an eine Amazon EC2 EC2-Instance anzuhängen. Diese Berechtigung wird von Amazon ECS benötigt, um Amazon EBS-Volumes an die Amazon EC2 EC2-Instance anzuhängen, die die zugehörige Amazon ECS-Aufgabe hostet.
- `ec2:DescribeVolume`— Ermöglicht einem Principal das Abrufen von Informationen über Amazon EBS-Volumes. Diese Berechtigung ist erforderlich, um den Lebenszyklus von Amazon EBS-Volumes zu verwalten.
- `ec2:DescribeAvailabilityZones`— Ermöglicht es einem Principal, Informationen über Availability Zones in Ihrem Konto abzurufen. Dies ist erforderlich, um den Lebenszyklus von EBS-Volumes zu verwalten.
- `ec2:DetachVolume`— Ermöglicht einem Principal, ein Amazon EBS-Volume von einer Amazon EC2 EC2-Instance zu trennen. Diese Berechtigung wird von Amazon ECS benötigt, um das Amazon EBS-Volume von der Amazon EC2 EC2-Instance zu trennen, die die zugehörige Amazon ECS-Aufgabe hostet, wenn die Aufgabe beendet wird.
- `ec2:DeleteVolume`— Ermöglicht einem Principal, ein Amazon EBS-Volume zu löschen. Diese Berechtigung wird von Amazon ECS benötigt, um Amazon EBS-Volumes zu löschen, die nicht mehr von der Amazon ECS-Aufgabe verwendet werden.
- `ec2:DeleteTags`— Ermöglicht einem Principal, das `AmazonECSManaged` Tag von einem Amazon EBS-Volume zu löschen. Diese Berechtigung wird von Amazon ECS benötigt, um den Zugriff auf ein Amazon EBS-Volume zu entfernen, nachdem es nicht mehr mit einem Amazon ECS-Workload verknüpft ist. Dies gilt nur, wenn ein Amazon EBS-Volume nach dem Herunterfahren der Aufgabe nicht gelöscht wird.

Es folgt eine Beispielrichtlinie `AmazonECSInfrastructureRolePolicyForVolumes`.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "CreateEBSManagedVolume",
   "Effect": "Allow",
   "Action": "ec2:CreateVolume",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "ArnLike": {
```

```
      "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
     },
     "StringEquals": {
      "aws:RequestTag/AmazonECSManaged": "true"
     }
    }
   },
   {
    "Sid": "TagOnCreateVolume",
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:*:*:volume/*",
    "Condition": {
     "ArnLike": {
      "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
     },
     "StringEquals": {
      "ec2:CreateAction": "CreateVolume",
      "aws:RequestTag/AmazonECSManaged": "true"
     }
    }
   },
   {
    "Sid": "DescribeVolumesForLifecycle",
    "Effect": "Allow",
    "Action": [
     "ec2:DescribeVolumes",
     "ec2:DescribeAvailabilityZones"
    ],
    "Resource": "*"
   },
   {
    "Sid": "ManageEBSVolumeLifecycle",
    "Effect": "Allow",
    "Action": [
     "ec2:AttachVolume",
     "ec2:DetachVolume"
    ],
    "Resource": "arn:aws:ec2:*:*:volume/*",
    "Condition": {
     "StringEquals": {
      "aws:ResourceTag/AmazonECSManaged": "true"
     }
    }
```

```
  },
  {
   "Sid": "ManageVolumeAttachmentsForEC2",
   "Effect": "Allow",
   "Action": [
    "ec2:AttachVolume",
    "ec2:DetachVolume"
   ],
   "Resource": "arn:aws:ec2:*:*:instance/*"
  },
  {
   "Sid": "DeleteEBSManagedVolume",
   "Effect": "Allow",
   "Action": "ec2:DeleteVolume",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "ArnLike": {
     "aws:ResourceTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
    },
    "StringEquals": {
     "aws:ResourceTag/AmazonECSManaged": "true"
    }
   }
  }
 ]
}
```

Amazon EC2 EC2-Rolle ContainerServicefor

Amazon ECS fügt diese Richtlinie einer Service-Rolle zu, mit der Amazon ECS in Ihrem Namen Aktionen gegen Amazon-EC2-Instances oder externe Instances ausführen kann.

Diese Richtlinie gewährt Administratorberechtigungen, die es Amazon ECS-Container-Instances ermöglichen, in AWS Ihrem Namen Aufrufe zu tätigen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

Überlegungen

Beachten Sie die folgenden Empfehlungen und Überlegungen bei der Verwendung der durch `AmazonEC2ContainerServiceforEC2Role` verwalteten IAM-Richtlinie.

- Wenn Sie den standardmäßigen Sicherheitshinweisen zur Erteilung von geringsten Privilegien folgen, können Sie die von `AmazonEC2ContainerServiceforEC2Role` verwaltete Richtlinie

modifizieren, um Ihren spezifischen Anforderungen gerecht zu werden. Wenn eine der in der verwalteten Richtlinie erteilten Berechtigungen für Ihren Anwendungsfall nicht benötigt wird, erstellen Sie eine benutzerdefinierte Richtlinie und fügen Sie nur die erforderlichen Berechtigungen hinzu. Zum Beispiel wird die `UpdateContainerInstancesState`-Berechtigung für die Entleerung von Spot-Instances bereitgestellt. Wenn diese Berechtigung für Ihren Anwendungsfall nicht benötigt wird, schließen Sie sie mit einer benutzerdefinierten Richtlinie aus. Weitere Informationen finden Sie unter Details zu Berechtigungen.

- Container, die auf Ihren Container-Instances ausgeführt werden, haben Zugriff auf alle Berechtigungen, die der Container-Instance-Rolle über Instance-Metadaten zur Verfügung gestellt werden. Wir empfehlen Ihnen, die Berechtigungen in Ihrer Container-Instance-Rolle auf die minimale Liste von Berechtigungen, die in der verwalteten Richtlinie `AmazonEC2ContainerServiceforEC2Role` bereitgestellt werden, zu begrenzen. Wenn die Container in Ihren Aufgaben besondere Berechtigungen erfordern, die nicht aufgelistet sind, empfehlen wir, für diese Aufgaben ihre eigenen IAM-Rollen bereitzustellen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

  Sie können Container auf der `docker0`-Bridge vom Zugriff auf die Berechtigungen, die der Container-Instance-Rolle bereitgestellt wurden, abhalten. Sie können dies tun, während Sie weiterhin die Berechtigungen zulassen, die von IAM-Rolle für Amazon ECS-Aufgaben durch ausführen des folgenden Befehlsiptables auf Ihren Container-Instances bereitgestellt werden. Container können Instance-Metadaten mit dieser Regel nicht abfragen. Dieser Befehl setzt die Standard-Docker-Bridge-Konfiguration voraus und funktioniert nicht für Container, die den `host`-Netzwerkmodus verwenden. Weitere Informationen finden Sie unter Netzwerkmodus.

  ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER USER 1 --in-interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Sie müssen diese iptables-Regel auf Ihrer Container-Instance speichern, damit sie einen Neustart übersteht. Verwenden Sie für die Amazon-ECS-optimierte AMI den folgenden Befehl. Informationen über andere Betriebssysteme finden Sie in der Dokumentation für dieses Betriebssystem.

  - Für das Amazon-ECS-optimierte Amazon Linux 2-AMI:

    ```
    sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now iptables
    ```

  - Für das Amazon-ECS-optimierte Amazon Linux AMI:

```
sudo service iptables save
```

Details zu Berechtigungen

Die von `AmazonEC2ContainerServiceforEC2Role` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Nach den standardmäßigen Sicherheitshinweisen zur Erteilung von geringsten Privilegien kann die von `AmazonEC2ContainerServiceforEC2Role` verwaltete Richtlinie als Leitfaden verwendet werden. Wenn Sie eine der in der verwalteten Richtlinie erteilten Berechtigungen für Ihren Anwendungsfall nicht benötigen, erstellen Sie eine benutzerdefinierte Richtlinie und fügen Sie nur die erforderlichen Berechtigungen hinzu.

- `ec2:DescribeTags`: Ermöglicht es einem Prinzipal, die Tags zu beschreiben, die einer Amazon-EC2-Instance zugeordnet sind. Diese Berechtigung wird vom Amazon-ECS-Container-Agent verwendet, um die Weitergabe von Ressourcen-Tags zu unterstützen. Weitere Informationen finden Sie unter So werden Ressourcen markiert.
- `ecs:CreateCluster`: Ermöglicht es einem Prinzipal, einen Amazon-ECS-Cluster zu erstellen. Diese Berechtigung wird vom Amazon-ECS-Container-Agenten verwendet, um einen `default`-Cluster zu erstellen, falls noch keiner vorhanden ist.
- `ecs:DeregisterContainerInstance`: Ermöglicht es einem Prinzipal, eine Amazon-ECS-Container-Instance von einem Cluster abzumelden. Der Amazon ECS-Container-Agent ruft diesen API-Vorgang nicht auf, aber diese Berechtigung bleibt bestehen, um die Abwärtskompatibilität sicherzustellen.
- `ecs:DiscoverPollEndpoint`: Diese Aktion gibt Endpunkte zurück, die der Amazon-ECS-Container-Agent zur Abfrage von Aktualisierungen verwendet.
- `ecs:Poll`: Ermöglicht dem Amazon-ECS-Container-Agent die Kommunikation mit der Amazon-ECS-Steuerungsebene, um Änderungen des Aufgabenzustands zu melden.
- `ecs:RegisterContainerInstance`: Ermöglicht es einem Prinzipal, eine Container-Instance bei einem Cluster zu registrieren. Diese Berechtigung wird vom Amazon ECS-Container-Agenten verwendet, um die Amazon EC2 EC2-Instance bei einem Cluster zu registrieren und die Weitergabe von Ressourcen-Tags zu unterstützen.
- `ecs:StartTelemetrySession`: Ermöglicht dem Amazon-ECS-Container-Agent die Kommunikation mit der Amazon-ECS-Steuerungsebene, um Integritätsinformationen und Metriken für jeden Container und jede Aufgabe zu melden.

- `ecs:TagResource` – Ermöglicht dem Amazon-ECS-Container-Agenten, Cluster bei der Erstellung mit Tags zu versehen und Tags zu Container-Instances hinzuzufügen, wenn sie in einem Cluster registriert werden.
- `ecs:UpdateContainerInstancesState`: Ermöglicht es einem Prinzipal, den Status einer Amazon-ECS-Container-Instance zu ändern. Diese Berechtigung wird vom Amazon-ECS-Container-Agent für Spot-Instance Draining verwendet.
- `ecs:Submit*`: Dies umfasst die API-Aktionen `SubmitAttachmentStateChanges`, `SubmitContainerStateChange` und `SubmitTaskStateChange`. Sie werden vom Amazon-ECS-Container-Agent verwendet, um Statusänderungen für jede Ressource an die Amazon-ECS-Steuerungsebene zu melden. Die `SubmitContainerStateChange` Berechtigung wird nicht mehr vom Amazon ECS-Container-Agenten verwendet, sondern dient weiterhin dazu, die Abwärtskompatibilität sicherzustellen.
- `ecr:GetAuthorizationToken`: Ermöglicht einem Prinzipal, ein Autorisierungstoken abzurufen. Ein Autorisierungs-Token stellt Ihre Anmeldeinformationen zur IAM-Authentifizierung dar und kann verwendet werden, um auf jede Amazon ECR-Registrierung zuzugreifen, auf die das IAM-Prinzipal Zugriff hat. Das erhaltene Autorisierungs-Token ist 12 Stunden gültig.
- `ecr:BatchCheckLayerAvailability`: Wenn ein Container-Image in ein privates Amazon ECR-Repository gesendet wird, wird bei jeder Image-Ebene überprüft, ob es bereits gepusht wurde. Wenn dies der Fall ist, wird die Image-Ebene übersprungen.
- `ecr:GetDownloadUrlForLayer`: Wenn ein Container-Image aus einem privaten Amazon ECR-Repository abgerufen wird, wird diese API für jede Image-Layer, die noch nicht zwischengespeichert ist, einmal aufgerufen.
- `ecr:BatchGetImage`: Wenn ein Container-Image aus einem privaten Amazon ECR-Repository abgerufen wird, wird diese API einmal aufgerufen, um das Image-Manifest abzurufen.
- `logs:CreateLogStream`— Ermöglicht einem Prinzipal, einen CloudWatch Logs-Log-Stream für eine angegebene Protokollgruppe zu erstellen.
- `logs:PutLogEvents`: Ermöglicht es einem Prinzipal, einen Stapel von Protokollereignissen in einen angegebenen Protokoll-Stream hochzuladen.

Es folgt eine Beispielrichtlinie `AmazonEC2ContainerServiceforEC2Role`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": [
                        "CreateCluster",
                        "RegisterContainerInstance"
                    ]
                }
            }
        }
    ]
}
```

Amazon EC2 ContainerService EventsRole

Diese Richtlinie gewährt Berechtigungen, die es Amazon EventBridge (ehemals CloudWatch Events) ermöglichen, Aufgaben in Ihrem Namen auszuführen. Diese Richtlinie kann der IAM-Rolle zugeordnet werden, die beim Erstellen geplanter Tasks angegeben wird. Weitere Informationen finden Sie unter Amazon ECS EventBridge IAM-Rolle.

Details zu Berechtigungen

Diese Richtlinie umfasst die folgenden Berechtigungen.

- `ecs`— Ermöglicht einem Principal in einem Service, die Amazon RunTask ECS-API aufzurufen. Ermöglicht es einem Principal in einem Service, Tags (`TagResource`) hinzuzufügen, wenn er die Amazon RunTask ECS-API aufruft.
- `iam`: Ermöglicht das Übergeben einer beliebigen IAM-Service-Rolle an alle Amazon-ECS-Aufgaben.

Es folgt eine Beispielrichtlinie `AmazonEC2ContainerServiceEventsRole`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

AmazonECS TaskExecution RolePolicy

Die `AmazonECSTaskExecutionRolePolicy` verwaltete IAM-Richtlinie gewährt die Berechtigungen, die der Amazon ECS-Container-Agent und die AWS Fargate Container-Agenten benötigen, um AWS API-Aufrufe in Ihrem Namen durchzuführen. Diese Richtlinie kann Ihrer IAM-Rolle zur Aufgabenausführung hinzugefügt werden. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Details zu Berechtigungen

Die von `AmazonECSTaskExecutionRolePolicy` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Nach den standardmäßigen Sicherheitshinweisen zur Erteilung von geringsten Privilegien kann die von `AmazonECSTaskExecutionRolePolicy` verwaltete Richtlinie als Leitfaden verwendet werden. Wenn eine der Berechtigungen, die in der verwalteten Richtlinie erteilt werden, für Ihren Anwendungsfall nicht erforderlich ist, erstellen Sie eine benutzerdefinierte Richtlinie, und fügen Sie nur die erforderlichen Berechtigungen hinzu.

- `ecr:GetAuthorizationToken`: Ermöglicht einem Prinzipal, ein Autorisierungstoken abzurufen. Ein Autorisierungs-Token stellt Ihre Anmeldeinformationen zur IAM-Authentifizierung dar und kann verwendet werden, um auf jede Amazon ECR-Registrierung zuzugreifen, auf die das IAM-Prinzipal Zugriff hat. Das erhaltene Autorisierungs-Token ist 12 Stunden gültig.
- `ecr:BatchCheckLayerAvailability`: Wenn ein Container-Image in ein privates Amazon ECR-Repository gesendet wird, wird bei jeder Image-Ebene überprüft, ob es bereits gepusht wurde. Wenn es gepusht wird, wird die Image-Ebene übersprungen.
- `ecr:GetDownloadUrlForLayer`: Wenn ein Container-Image aus einem privaten Amazon ECR-Repository abgerufen wird, wird diese API für jede Image-Layer, die noch nicht zwischengespeichert ist, einmal aufgerufen.
- `ecr:BatchGetImage`: Wenn ein Container-Image aus einem privaten Amazon ECR-Repository abgerufen wird, wird diese API einmal aufgerufen, um das Image-Manifest abzurufen.
- `logs:CreateLogStream`— Ermöglicht einem Prinzipal, einen CloudWatch Logs-Log-Stream für eine angegebene Protokollgruppe zu erstellen.
- `logs:PutLogEvents`: Ermöglicht es einem Prinzipal, einen Stapel von Protokollereignissen in einen angegebenen Protokoll-Stream hochzuladen.

Es folgt eine Beispielrichtlinie `AmazonECSTaskExecutionRolePolicy`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

AmazonECS-Richtlinie ServiceRole

Die `AmazonECSServiceRolePolicy`-verwaltete IAM-Richtlinie ermöglicht es Amazon Elastic Container Service, Ihren Cluster zu verwalten. Diese Richtlinie kann Ihrer IAM-Rolle zur Aufgabenausführung hinzugefügt werden. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Details zu Berechtigungen

Die von `AmazonECSServiceRolePolicy` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Nach den standardmäßigen Sicherheitshinweisen zur Erteilung von geringsten Privilegien kann die von `AmazonECSServiceRolePolicy` verwaltete Richtlinie als Leitfaden verwendet werden. Wenn eine der Berechtigungen, die in der verwalteten Richtlinie erteilt werden, für Ihren Anwendungsfall nicht erforderlich ist, erstellen Sie eine benutzerdefinierte Richtlinie, und fügen Sie nur die erforderlichen Berechtigungen hinzu.

- `autoscaling`: Ermöglicht es Prinzipalen, Amazon EC2 Auto Scaling Ressourcen zu erstellen, zu verwalten und zu beschreiben. Dies ist erforderlich, wenn Sie Amazon EC2 Auto Scaling Scaling-Gruppen verwalten und die Cluster-Auto-Scaling-Funktion verwenden.
- `autoscaling-plans` – Ermöglicht es Prinzipalen, Auto-Scaling-Pläne zu erstellen, zu löschen und zu beschreiben.
- `cloudwatch`— Ermöglicht es Prinzipalen, CloudWatch Amazon-Alarme zu erstellen, zu verwalten und zu beschreiben.

- `ec2`— Ermöglicht Principals, Amazon EC2 EC2-Instances auszuführen und Netzwerkschnittstellen und Tags zu erstellen und zu verwalten.
- `elasticloadbalancing`: Ermöglicht Prinzipalen das Erstellen, Beschreiben und Löschen von Elastic Load Balancing-Lastenausgleichsdiensten. Principals werden auch in der Lage sein, Zielgruppen hinzuzufügen und zu beschreiben.
- `logs`— Ermöglicht Prinzipalen das Erstellen und Beschreiben von Amazon CloudWatch Logs-Protokollgruppen. Prinzipale können auch Protokollereignisse für diese Protokollgruppen auflisten.
- `route53`: Ermöglicht Prinzipalen das Erstellen, Verwalten und Löschen von gehosteten Amazon Route 53-Zonen. Prinzipale können auch die Konfiguration und Informationen zur Amazon Route 53 Integritätsprüfung anzeigen. Weitere Informationen über gehostete Zonen finden Sie unter Arbeiten mit gehosteten Zonen.
- `servicediscovery`— Ermöglicht Prinzipalen, AWS Cloud Map Dienste zu erstellen, zu verwalten und zu löschen und private DNS-Namespaces zu erstellen.
- `events`— Ermöglicht Prinzipalen das Erstellen, Verwalten und Löschen von EventBridge Amazon-Regeln und deren Zielen.

Es folgt eine Beispielrichtlinie `AmazonECSServiceRolePolicy`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECSTaskManagement",
            "Effect": "Allow",
            "Action": [
                "ec2:AttachNetworkInterface",
                "ec2:CreateNetworkInterface",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:Describe*",
                "ec2:DetachNetworkInterface",
                "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                "elasticloadbalancing:DeregisterTargets",
                "elasticloadbalancing:Describe*",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "route53:ChangeResourceRecordSets",
                "route53:CreateHealthCheck",
```

```
                "route53:DeleteHealthCheck",
                "route53:Get*",
                "route53:List*",
                "route53:UpdateHealthCheck",
                "servicediscovery:DeregisterInstance",
                "servicediscovery:Get*",
                "servicediscovery:List*",
                "servicediscovery:RegisterInstance",
                "servicediscovery:UpdateInstanceCustomHealthStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AutoScaling",
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AutoScalingManagement",
            "Effect": "Allow",
            "Action": [
                "autoscaling:DeletePolicy",
                "autoscaling:PutScalingPolicy",
                "autoscaling:SetInstanceProtection",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:PutLifecycleHook",
                "autoscaling:DeleteLifecycleHook",
                "autoscaling:CompleteLifecycleAction",
                "autoscaling:RecordLifecycleActionHeartbeat"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "autoscaling:ResourceTag/AmazonECSManaged": "false"
                }
            }
        },
        {
            "Sid": "AutoScalingPlanManagement",
            "Effect": "Allow",
            "Action": [
```

```
                "autoscaling-plans:CreateScalingPlan",
                "autoscaling-plans:DeleteScalingPlan",
                "autoscaling-plans:DescribeScalingPlans",
                "autoscaling-plans:DescribeScalingPlanResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EventBridge",
            "Effect": "Allow",
            "Action": [
                "events:DescribeRule",
                "events:ListTargetsByRule"
            ],
            "Resource": "arn:aws:events:*:*:rule/ecs-managed-*"
        },
        {
            "Sid": "EventBridgeRuleManagement",
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:PutTargets"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:ManagedBy": "ecs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CWAlarmManagement",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm"
            ],
            "Resource": "arn:aws:cloudwatch:*:*:alarm:*"
        },
        {
            "Sid": "ECSTagging",
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*"
        },
        {
            "Sid": "CWLogGroupManagement",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/ecs/*"
        },
        {
            "Sid": "CWLogStreamManagement",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/ecs/*:log-stream:*"
        },
        {
            "Sid": "ExecuteCommandSessionManagement",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ExecuteCommand",
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ecs:*:*:task/*",
                "arn:aws:ssm:*:*:document/AmazonECS-ExecuteInteractiveCommand"
            ]
        },
        {
```

```
            "Sid": "CloudMapResourceCreation",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:CreateHttpNamespace",
                "servicediscovery:CreateService"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "AmazonECSManaged"
                    ]
                }
            }
        },
        {
            "Sid": "CloudMapResourceTagging",
            "Effect": "Allow",
            "Action": "servicediscovery:TagResource",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/AmazonECSManaged": "*"
                }
            }
        },
        {
            "Sid": "CloudMapResourceDeletion",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:DeleteService"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AmazonECSManaged": "false"
                }
            }
        },
        {
            "Sid": "CloudMapResourceDiscovery",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:DiscoverInstances",
```

```
                "servicediscovery:DiscoverInstancesRevision"
            ],
            "Resource": "*"
        }
    ]
}
```

AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecur

Bietet Administratorzugriff auf Secrets Manager und andere AWS Services AWS Private Certificate Authority, die zur Verwaltung der TLS-Funktionen von Amazon ECS Service Connect in Ihrem Namen erforderlich sind.

Details zu Berechtigungen

Die von `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` verwaltete IAM-Richtlinie muss die folgende Berechtigungen umfassen. Nach den standardmäßigen Sicherheitshinweisen zur Erteilung von geringsten Privilegien kann die von `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` verwaltete Richtlinie als Leitfaden verwendet werden. Wenn eine der Berechtigungen, die in der verwalteten Richtlinie erteilt werden, für Ihren Anwendungsfall nicht erforderlich ist, erstellen Sie eine benutzerdefinierte Richtlinie, und fügen Sie nur die erforderlichen Berechtigungen hinzu.

- `secretsmanager:CreateSecret`— Ermöglicht dem Principal, das Geheimnis zu erstellen. Es ist für Service Connect TLS erforderlich. Amazon ECS hält den privaten Schlüssel des Kunden im Secrets Manager des Kunden geheim.
- `secretsmanager:TagResource`— Ermöglicht dem Principal, dem erstellten Geheimnis ein Tag zuzuweisen. Es ist für Service Connect TLS erforderlich, da Amazon ECS das Geheimnis im Namen des Kunden erstellt und das Tag mit der Ressource verknüpft. Diese Tags bieten dem Kunden eine einfachere Möglichkeit, den verwalteten geheimen Schlüssel zu identifizieren und Aktionen im Zusammenhang mit diesen Geheimnissen einzuschränken.
- `secretsmanager:DescribeSecret`— Erlauben Sie dem Principal, das Geheimnis zu beschreiben und die aktuelle Version abzurufen. Es ist erforderlich, dass Amazon ECS die Materialrotation mit Amazon ECS Service Connect TLS durchführt.
- `secretsmanager:UpdateSecret`— Erlaubt dem Principal, das Geheimnis zu aktualisieren. Amazon ECS muss die Amazon ECS Service Connect TLS-Materialrotation durchführen und das Geheimnis mit neuen Materialien aktualisieren.

- `secretsmanager:GetSecretValue`— Erlaubt dem Principal, den geheimen Wert abzurufen. Es ist erforderlich, dass Amazon ECS die Materialrotation mit Amazon ECS Service Connect TLS durchführt.
- `secretsmanager:PutSecretValue`— Erlaubt dem Principal, den geheimen Wert einzugeben. Es ist erforderlich, dass Amazon ECS die Materialrotation mit Amazon ECS Service Connect TLS durchführt.
- `secretsmanager:UpdateSecretVersionStage`— Erlaubt dem Principal, die geheime Versionsphase zu aktualisieren. Es ist erforderlich, dass Amazon ECS die Materialrotation mit Amazon ECS Service Connect TLS durchführt.
- `acm-pca:IssueCertificate`— Erlaubt dem Principal, Amazon ECS Service Connect TLS anzufordern IssueCertificate. `End entity certificate` ECS musste ein Zertifikat für den Upstream-Service des Kunden generieren.
- `acm-pca:GetCertificate`— Erlaubt dem Principal, Amazon ECS Service Connect TLS anzufordern GetCertificate . `End entity certificate`
- `acm-pca:GetCertificateAuthorityCertificate`— Erlaubt dem Prinzipal, das Zertifikat der Zertifizierungsstelle abzurufen. Es ist für Amazon ECS Service Connect TLS erforderlich, damit der Downstream-Service des Kunden dem Upstream-Endentitätszertifikat vertrauen kann.
- `acm-pca:DescribeCertificateAuthority`— Erlaubt dem Principal, Details über die Zertifizierungsstelle abzurufen. Amazon ECS Service Connect TLS muss Informationen wie den Signaturalgorithmus zur Erstellung der CSR (Certificate Signing Request) wiederverwenden.

Es folgt eine Beispielrichtlinie `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:CreateSecret",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                    "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
```

```
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "TagOnCreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:TagResource",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                     "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "RotateTLSCertificateSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:DeleteSecret",
                "secretsmanager:RotateSecret",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
"ecs-sc",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
```

```
                }
            }
        },
        {
            "Sid": "ManagePrivateCertificateAuthority",
            "Effect": "Allow",
            "Action": [
                "acm-pca:GetCertificate",
                "acm-pca:GetCertificateAuthorityCertificate",
                "acm-pca:DescribeCertificateAuthority"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true"
                }
            }
        },
        {
            "Sid": "ManagePrivateCertificateAuthorityForIssuingEndEntityCertificate",
            "Effect": "Allow",
            "Action": [
                "acm-pca:IssueCertificate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true",
                    "acm-pca:TemplateArn":"arn:aws:acm-pca:::template/
EndEntityCertificate/V1"
                }
            }
        }
    ]
}
```

AWSApplicationAutoscalingECSServicePolicy

Sie können `AWSApplicationAutoscalingECSServicePolicy` nicht an Ihre IAM-Entitäten anhängen. Diese Richtlinie ist einer serviceverknüpften Rolle zugeordnet, die die Application Auto Scaling in Ihrem Namen ermöglicht. Weitere Informationen finden Sie unter Serviceverknüpfte Rollen für Application Auto Scaling.

AWSCodeDeployRoleForECS

Sie können `AWSCodeDeployRoleForECS` nicht an Ihre IAM-Entitäten anhängen. Diese Richtlinie ist mit einer dienstbezogenen Rolle verknüpft, die es CodeDeploy ermöglicht, Aktionen in Ihrem Namen durchzuführen. Weitere Informationen finden Sie unter Erstellen einer Servicerolle für CodeDeploy im AWS CodeDeploy Benutzerhandbuch.

AWSCodeDeployRoleForECSLimited

Sie können `AWSCodeDeployRoleForECSLimited` nicht an Ihre IAM-Entitäten anhängen. Diese Richtlinie ist mit einer dienstbezogenen Rolle verknüpft, mit der CodeDeploy Sie Aktionen in Ihrem Namen ausführen können. Weitere Informationen finden Sie unter Erstellen einer Servicerolle für CodeDeploy im AWS CodeDeploy Benutzerhandbuch.

Amazon ECS-Updates für AWS verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amazon ECS an, seit dieser Service begonnen hat, diese Änderungen zu verfolgen. Um automatische Warnungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Amazon-ECS-Dokumentverlauf-Seite.

Änderung	Beschreibung	Datum
Neue InfrastructureRolePolicyForServiceConnectTransportLayerAmazonECS-Sicherheitsrichtlinie hinzufügen	Es wurde eine neue InfrastructureRolePolicyForServiceConnectTransportLayerSecurity AmazonECS-Richtlinie hinzugefügt, die administrativen Zugriff auf AWS KMS AWS Private Certificate Authority, Secrets Manager bietet und dafür sorgt, dass die TLS-Funktionen von AmazonECS Service Connect ordnungsgemäß funktionieren.	22. Januar 2024

Änderung	Beschreibung	Datum
Neue Richtlinie hinzufügen n AmazonECS Volumes InfrastructureRole PolicyFor	Die `AmazonECSInfrastructureRolePolicyForVolumes` Richtlinie wurde hinzugefügt. Die Richtlinie gewährt die Berechtigungen, die Amazon ECS benötigt, um AWS API-Aufrufe zur Verwaltung von Amazon EBS-Volumes im Zusammenhang mit Amazon ECS-Workloads durchzuführen.	11. Januar 2024
Berechtigungen zur AmazonECS-Richtlinie ServiceRole hinzufügen	Die `AmazonECSServiceRolePolicy` verwaltete IAM-Richtlinie wurde mit neuen `autoscaling` und `autoscaling-plans` zusätzlichen `events` Berechtigungen aktualisiert.	4. Dezember 2023
Fügen Sie Berechtigungen zu AmazonEC2 hinzu Container Service EventsRole	Die `AmazonECSServiceRolePolicy` verwaltete IAM-Richtlinie wurde aktualisiert, um den Zugriff auf den API-Vorgang zu ermöglichen. AWS Cloud Map `DiscoverInstancesRevision`	04. Oktober 2023

Änderung	Beschreibung	Datum
Fügen Sie Berechtigungen zu ContainerServiceforAmazonEC2 EC2Role hinzu	Die `AmazonEC2ContainerServiceforEC2Role` Richtlinie wurde geändert, um die `ecs:TagResource` Berechtigung hinzuzufügen. Dazu gehört auch eine Bedingung, die die Berechtigung nur auf neu erstellte Cluster und registrierte Container-Instances beschränkt.	6. März 2023
Berechtigungen zu the section called “AmazonECS_FullAccess” hinzufügen	Die `AmazonECS_FullAccess` Richtlinie wurde dahingehend geändert, dass die `elasticloadbalancing:AddTags` Berechtigung hinzugefügt wurde. Dazu gehört auch eine Bedingung, die die Berechtigung nur auf neu erstellte Load Balancer, Zielgruppen, Regeln und neu erstellte Listener beschränkt. Diese Berechtigung erlaubt nicht das Hinzufügen von Tags zu bereits erstellten Elastic-Load-Balancing-Ressourcen.	4. Januar 2023
Amazon ECS hat mit der Verfolgung von Änderungen begonnen	Amazon ECS hat damit begonnen, Änderungen für seine AWS verwalteten Richtlinien nachzuverfolgen.	8. Juni 2021

Schrittweise Einstellung der AWS verwalteten IAM-Richtlinien für Amazon Elastic Container Service

Die folgenden AWS verwalteten IAM-Richtlinien werden schrittweise eingestellt. Diese Richtlinien werden nun durch die aktualisierten Richtlinien ersetzt. Wir empfehlen Ihnen, Ihre Benutzer oder Rollen zu aktualisieren, damit diese die aktualisierten Richtlinien verwenden.

Amazon EC2 ContainerService FullAccess

> **Important**
>
> Die von `AmazonEC2ContainerServiceFullAccess` verwaltete IAM-Richtlinie wurde am 29 .Januar 2021 als Reaktion auf eine Sicherheitsfeststellung mit der `iam:passRole`-Berechtigung ausgemustert. Diese Berechtigung gewährt den Zugriff auf alle Ressourcen, einschließlich der Anmeldeinformationen für Rollen im Konto. Da die Richtlinie nun schrittweise beendet ist, können Sie sie keinen neuen Benutzern oder Rollen anfügen. Alle Benutzer oder Rollen, denen die Richtlinie bereits angefügt ist, können sie auch weiterhin verwenden. Wir empfehlen jedoch, dass Sie Ihre Benutzer oder Rollen so aktualisieren, dass stattdessen die von `AmazonECS_FullAccess` verwaltete Richtlinie verwendet wird. Weitere Informationen finden Sie unter Migration zur von `AmazonECS_FullAccess` verwalteten Richtlinie.

AmazonEC2-Rolle ContainerService

> **Important**
>
> Die von `AmazonEC2ContainerServiceRole` verwaltete IAM-Richtlinie wird ausgemustert. Sie wird jetzt durch die servicegebundene Amazon-ECS-Rolle ersetzt. Weitere Informationen finden Sie unter Verwendung von serviceverknüpften Rollen für Amazon ECS.

Amazon EC2 ContainerService AutoscaleRole

> **Important**
>
> Die von `AmazonEC2ContainerServiceAutoscaleRole` verwaltete IAM-Richtlinie wird ausgemustert. Sie wird jetzt durch das serviceverknüpfte Application Auto Scaling für Amazon

ECS ersetzt. Weitere Informationen finden Sie unter Serviceverknüpfte Rollen für Application Auto Scaling im Benutzerhandbuch zu Application Auto Scaling.

Migration zur von **`AmazonECS_FullAccess`** verwalteten Richtlinie

Die von `AmazonEC2ContainerServiceFullAccess` verwaltete IAM-Richtlinie wurde am 29 .Januar 2021 als Reaktion auf eine Sicherheitsfeststellung mit der `iam:passRole`-Berechtigung ausgemustert. Diese Berechtigung gewährt den Zugriff auf alle Ressourcen, einschließlich der Anmeldeinformationen für Rollen im Konto. Da die Richtlinie nun schrittweise beendet ist, können Sie die Richtlinie keinen neuen Gruppen, Benutzern oder Rollen anfügen. Alle Gruppen, Benutzer oder Rollen, denen die Richtlinie bereits zugeordnet ist, können sie weiterhin verwenden. Wir empfehlen jedoch, dass Sie Ihre Gruppen, Benutzer oder Rollen so aktualisieren, dass stattdessen die von `AmazonECS_FullAccess` verwaltete Richtlinie verwendet wird.

Die Berechtigungen, die von der `AmazonECS_FullAccess`-Richtlinie gewährt werden, enthalten die vollständige Liste der Berechtigungen, die für die Verwendung von ECS als Administrator erforderlich sind. Wenn Sie derzeit Berechtigungen verwenden, die durch die `AmazonEC2ContainerServiceFullAccess` Richtlinie gewährt wurden und nicht in der `AmazonECS_FullAccess` Richtlinie enthalten sind, können Sie sie zu einer Inline-Richtlinienerklärung hinzufügen. Weitere Informationen finden Sie unter AWS verwaltete Richtlinien für Amazon Elastic Container Service.

Verwenden Sie die folgenden Schritte, um festzustellen, ob Sie über Gruppen, Benutzer oder Rollen verfügen, die derzeit die `AmazonEC2ContainerServiceFullAccess`-verwaltete IAM-Richtlinie verwenden. Aktualisieren Sie sie dann, um die frühere Richtlinie zu trennen, und die `AmazonECS_FullAccess`-Richtlinie anzufügen.

Um eine Gruppe, einen Benutzer oder eine Rolle zur Verwendung der FullAccess AmazonECS_-Richtlinie () zu aktualisieren AWS Management Console

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien und suchen sie nach und wählen Sie die `AmazonEC2ContainerServiceFullAccess`-Richtlinie.
3. Wählen Sie die Registerkarte Verwendung von Richtlinien, die jede IAM-Rolle anzeigt, die derzeit diese Richtlinie verwendet.

4. Wählen Sie für jede IAM-Rolle, die derzeit die `AmazonEC2ContainerServiceFullAccess` Richtlinie verwendet, die Rolle aus und führen Sie die folgenden Schritte aus, um die auslaufende Richtlinie zu trennen und die Richtlinie anzuhängen. `AmazonECS_FullAccess`

   a. Wählen Sie auf der Registerkarte Berechtigungen das X neben der AmazonEC2-Richtlinie aus. ContainerService FullAccess

   b. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus.

   c. Wählen Sie Bestehende Richtlinien direkt anhängen, suchen Sie nach der FullAccessAmazonECS_-Richtlinie, wählen Sie sie aus und klicken Sie dann auf Weiter: Überprüfen.

   d. Prüfen Sie die Änderungen und wählen Sie Add permissions (Berechtigungen hinzufügen) aus.

   e. Wiederholen Sie diese Schritte für jede Gruppe, jeden Benutzer oder jede Rolle, die die `AmazonEC2ContainerServiceFullAccess`-Richtlinie verwendet.

So aktualisieren Sie eine Gruppe, einen Benutzer oder eine Rolle für die Verwendung der **`AmazonECS_FullAccess`**-Richtlinie (AWS CLI)

1. Verwenden Sie den generate-service-last-accessed-detailsBefehl, um einen Bericht zu erstellen, der Einzelheiten darüber enthält, wann die auslaufende Richtlinie zuletzt angewendet wurde.

```
aws iam generate-service-last-accessed-details \
     --arn arn:aws:iam::aws:policy/AmazonEC2ContainerServiceFullAccess
```

   Beispielausgabe:

```
{
    "JobId": "32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE"
}
```

2. Verwenden Sie die Job-ID aus der vorherigen Ausgabe mit dem get-service-last-accessed-detailsBefehl, um den Bericht über den Dienst abzurufen, auf den zuletzt zugegriffen wurde. In diesem Bericht wird der Amazon-Ressourcenname (ARN) der IAM-Entitäten angezeigt, die die auslaufende Richtlinie zuletzt verwendet haben.

```
aws iam get-service-last-accessed-details \
      --job-id 32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE
```

3. Verwenden Sie einen der folgenden Befehle, um die `AmazonEC2ContainerServiceFullAccess`-Richtlinie von einer Gruppe, einem Benutzer oder einer Rolle zu trennen.

   - detach-group-policy
   - detach-role-policy
   - detach-user-policy

4. Verwenden Sie einen der folgenden Befehle, um die `AmazonECS_FullAccess`-Richtlinie einer Gruppe, einem Benutzer oder einer Rolle anzufügen.

   - attach-group-policy
   - attach-role-policy
   - attach-user-policy

Verwendung von serviceverknüpften Rollen für Amazon ECS

Amazon Elastic Container Service verwendet AWS Identity and Access Management (IAM) service-verknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon EKS verknüpft ist. Serviceverknüpfte Rollen werden von Amazon ECS vordefiniert und schließen alle Berechtigungen ein, die der Service zum Aufrufen anderer AWS -Services in Ihrem Namen erfordert.

Eine serviceverknüpfte Rolle macht die Einrichtung von Amazon ECS einfacher, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon ECS definiert die Berechtigungen seiner mit dem Service verbundenen Rollen, und sofern nicht anders definiert, kann nur Amazon ECS seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Informationen zu anderen Services, die serviceorientierte Rollen unterstützen, finden Sie unter AWS services that work with IAM (-Services, die mit IAM funktionieren). Suchen Sie nach den Services, für die Yes (Ja) in der Spalte Service-linked roles (Serviceorientierte Rollen) angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

Serviceverknüpfte Rollenberechtigungen für Amazon ECS

Amazon ECS verwendet die mit dem Service verknüpfte Rolle mit dem Namen AWSServiceRoleForECS.

Die AWSServiceRoleForECS servicebezogene Rolle vertraut darauf, dass die folgenden Dienste die Rolle übernehmen:

- `ecs.amazonaws.com`

Die Rollenberechtigungsrichtlinie mit dem Namen AmazonECS ServiceRolePolicy ermöglicht Amazon ECS, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: Bei Verwendung des `awsvpc`-Netzwerkmodus für Ihre Amazon-ECS-Aufgaben verwaltet Amazon ECS den Lebenszyklus der Elastic-Network-Schnittstellen, die mit der Aufgabe verknüpft sind. Dazu gehören auch Tags, die Amazon ECS zu Ihren Elastic-Network-Schnittstellen hinzufügt.
- Aktion: Wenn Sie einen Load Balancer mit Ihrem Amazon-ECS-Service verwenden, verwaltet Amazon ECS die Registrierung und Aufhebung der Registrierung von Ressourcen mit dem Load Balancer.
- Aktion: Wenn Sie Amazon ECS Service Discovery verwenden, verwaltet Amazon ECS die erforderliche Route 53 und die AWS Cloud Map Ressourcen, damit Service Discovery funktioniert.
- Aktion: Wenn Sie das Auto Scaling des Amazon-ECS-Service verwenden, verwaltet Amazon ECS die erforderlichen Auto-Scaling-Ressourcen.
- Aktion: Amazon ECS erstellt und verwaltet CloudWatch Alarme und Protokollstreams, die Sie bei der Überwachung Ihrer Amazon ECS-Ressourcen unterstützen.
- Aktion: Wenn Sie Amazon ECS Exec verwenden, verwaltet Amazon ECS die Berechtigungen, die zum Starten von Amazon-ECS-Exec-Sitzungen für Ihre Aufgaben erforderlich sind.
- Aktion: Bei Verwendung von Amazon ECS Service Connect verwaltet Amazon ECS die erforderlichen AWS Cloud Map -Ressourcen zur Verwendung des Features.
- Aktion: Bei Verwendung von Amazon-ECS-Kapazitätsanbietern verwaltet Amazon ECS die Berechtigungen, die zum Ändern der Auto-Scaling-Gruppe und ihrer Amazon-EC2-Instances erforderlich sind.

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Erstellen einer serviceverknüpften Rolle für Amazon ECS

In den meisten Fällen müssen Sie die serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie einen Cluster erstellen oder einen Service in der AWS Management Console, der oder der AWS API erstellen oder aktualisieren AWS CLI, erstellt Amazon ECS die serviceverknüpfte Rolle für Sie. Wenn Sie die AWSServiceRoleForECSRolle nach dem Erstellen eines Clusters nicht sehen, gehen Sie wie folgt vor, um das Problem zu beheben:

- Überprüfen und konfigurieren Sie die Berechtigungen, damit Amazon ECS eine serviceverknüpfte Rolle in Ihrem Namen erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.
- Versuchen Sie den Vorgang zur Clustererstellung erneut, oder erstellen Sie die serviceverknüpfte Rolle manuell.

  Sie können die IAM-Konsole verwenden, um die AWSServiceRoleForECSserviceverknüpfte Rolle zu erstellen. Erstellen Sie in der AWS CLI oder der AWS API eine dienstverknüpfte Rolle mit dem `ecs.amazonaws.com` Dienstnamen. Weitere Informationen finden Sie unter Erstellen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

> **Important**
>
> Diese serviceverknüpfte Rolle kann in Ihrem Konto erscheinen, wenn Sie eine Aktion in einem anderen Service abgeschlossen haben, der die von dieser Rolle unterstützten Features verwendet.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie einen Cluster erstellen oder einen Service erstellen oder aktualisieren, erstellt Amazon ECS die serviceverknüpfte Rolle erneut für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen, können Sie mit demselben IAM-Verfahren die Rolle erneut erstellen.

Bearbeiten einer serviceverknüpften Rolle für Amazon ECS

Amazon ECS erlaubt Ihnen nicht, die AWSServiceRoleForECS serviceverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollenname

nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Löschen einer serviceverknüpften Rolle für Amazon ECS

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird. Sie müssen jedoch die Ressourcen für Ihre serviceverknüpften Rolle zunächst bereinigen, bevor Sie sie manuell löschen können.

> **Note**
>
> Wenn der Amazon-ECS-Service die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt der Löschvorgang möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut.

So überprüfen Sie, ob die serviceverknüpfte Rolle über eine aktive Sitzung verfügt

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Rollen und dann den AWSServiceRoleForECS Namen aus (nicht das Kontrollkästchen).
3. Wählen Sie auf der Seite Summary Access Advisor und überprüfen Sie die letzten Aktivitäten auf die serviceverknüpfte Rolle.

> **Note**
>
> Wenn Sie sich nicht sicher sind, ob Amazon ECS die AWSServiceRoleForECS Rolle verwendet, können Sie versuchen, die Rolle zu löschen. Wenn der Service die Rolle verwendet, schlägt die Löschung fehl und Sie können die Regionen anzeigen, in denen die Rolle verwendet wird. Wenn die Rolle verwendet wird, müssen Sie warten, bis die Sitzung beendet wird, bevor Sie die Rolle löschen können. Die Sitzung für eine serviceverknüpfte Rolle können Sie nicht widerrufen.

So entfernen Sie Amazon ECS-Ressourcen, die von der AWSServiceRoleForECS serviceverknüpften Rolle verwendet werden

Sie müssen alle Amazon ECS-Cluster in allen AWS Regionen löschen, bevor Sie die AWSServiceRoleForECS Rolle löschen können.

1. Skalieren Sie alle Amazon-ECS-Services auf eine gewünschte Anzahl von 0 in allen Regionen, und löschen Sie die Services. Weitere Informationen finden Sie unter Aktualisieren eines Amazon ECS-Service mithilfe der Konsole und Löschen eines Amazon ECS-Service mithilfe der Konsole.
2. Erzwingen Sie die Deregistrierung aller Container-Instances aus allen Clustern in allen Regionen. Weitere Informationen finden Sie unter Abmeldung einer Amazon ECS-Container-Instance.
3. Löschen Sie alle Amazon-ECS-Cluster in allen Regionen. Weitere Informationen finden Sie unter Löschen eines Amazon ECS-Clusters.

So löschen Sie die serviceverknüpfte Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die AWS API AWS CLI, um die AWSServiceRoleForECS serviceverknüpfte Rolle zu löschen. Weitere Informationen finden Sie unter Löschen einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Unterstützte Regionen für serviceverknüpfte Rollen von Amazon ECS

Amazon ECS unterstützt die Verwendung von serviceverknüpften Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter AWS -Regionen und -Endpunkte.

IAM-Rollen für Amazon ECS

Eine IAM-Rolle ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. In Amazon ECS können Sie Rollen erstellen, um Berechtigungen für Amazon ECS-Ressourcen wie Container oder Services zu erteilen.

Die Rollen, die Amazon ECS benötigt, hängen vom Starttyp der Aufgabendefinition und den von Ihnen verwendeten Funktionen ab. Ermitteln Sie anhand der folgenden Tabelle, welche IAM-Rollen Sie für Amazon ECS benötigen.

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Aufgabenausführungsrolle	Diese Rolle ermöglicht es Amazon ECS, andere AWS Dienste in Ihrem Namen zu nutzen.	Ihre Aufgabe wird auf AWS Fargateoder auf externen Instances gehostet und: • ruft ein Container-Image aus einem privaten Amazon ECR-Repository ab. • ruft ein Container-Image aus einem privaten Amazon ECR-Repository in einem anderen Konto als dem Konto ab, das die Aufgabe ausführt. • sendet CloudWatch Container-Protokolle mithilfe des Protokolltreibers an `awslogs` Logs. Ihre Aufgabe wird entweder auf Amazon EC2-Instances AWS Fargate oder auf Amazon EC2 EC2-Instances gehostet und: • verwendet die private Registrie	IAM-Rolle für die Amazon-ECS-Aufgabenausführung

Rolle	Definition	Wenn erforderlich	Weitere Informationen
		rungsauthentifizierung. • verwendet Runtime Monitoring. • Die Aufgabendefinition verweist mithilfe von Secrets Manager Manager-Geheimnissen oder AWS Systems Manager Parameter Store-Parametern auf sensible Daten.	
Aufgabenrolle	Diese Rolle ermöglicht es Ihrem Anwendungscode (auf dem Container), andere AWS Dienste zu verwenden.	Ihre Anwendung greift auf andere AWS Dienste wie Amazon S3 zu.	IAM-Rolle für Amazon ECS-Aufgaben
Rolle für Container-Instances	Diese Rolle ermöglicht es Ihren EC2-Instances oder externen Instances, sich beim Cluster zu registrieren.	Ihre Aufgabe wird auf Amazon EC2 EC2-Instances oder einer externen Instance gehostet.	IAM-Rolle für Amazon-ECS-Container-Instance
Amazon ECS Anywhere Anywhere-Rolle	Diese Rolle ermöglicht Ihren externen Instances den Zugriff auf AWS APIs.	Ihre Aufgabe wird auf externen Instanzen gehostet.	IAM-Rolle in Amazon ECS Anywhere

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Amazon CodeDeploy ECS-Rolle	Diese Rolle ermöglicht CodeDeploy es, Aktualisierungen an Ihren Diensten vorzunehmen.	Sie verwenden den Bereitstellungstyp CodeDeploy Blau/Grün, um Dienste bereitzustellen.	Amazon ECS CodeDeploy IAM-Rolle
Amazon EventBridge ECS-Rolle	Diese Rolle ermöglicht EventBridge es, Aktualisierungen an Ihren Diensten vorzunehmen.	Sie verwenden die EventBridge Regeln und Ziele, um Ihre Aufgaben zu planen.	Amazon ECS EventBridge IAM-Rolle

Rolle	Definition	Wenn erforderlich	Weitere Informationen
Rolle in der Amazon ECS-Infrastruktur	Diese Rolle ermöglicht es Amazon ECS, die Infrastrukturressourcen in Ihren Clustern zu verwalten.	• Sie möchten Amazon EBS-Volumes an Ihre Amazon ECS-Aufgaben vom Starttyp Fargate oder EC2 anhängen. Die Infrastrukturrolle ermöglicht es Amazon ECS, Amazon EBS-Volumes für Ihre Aufgaben zu verwalten. • Sie möchten Transport Layer Security (TLS) verwenden, um den Verkehr zwischen Ihren Amazon ECS Service Connect-Services zu verschlüsseln.	IAM-Rolle für die Amazon ECS-Infrastruktur

Bewährte Methoden für IAM-Rollen in Amazon ECS

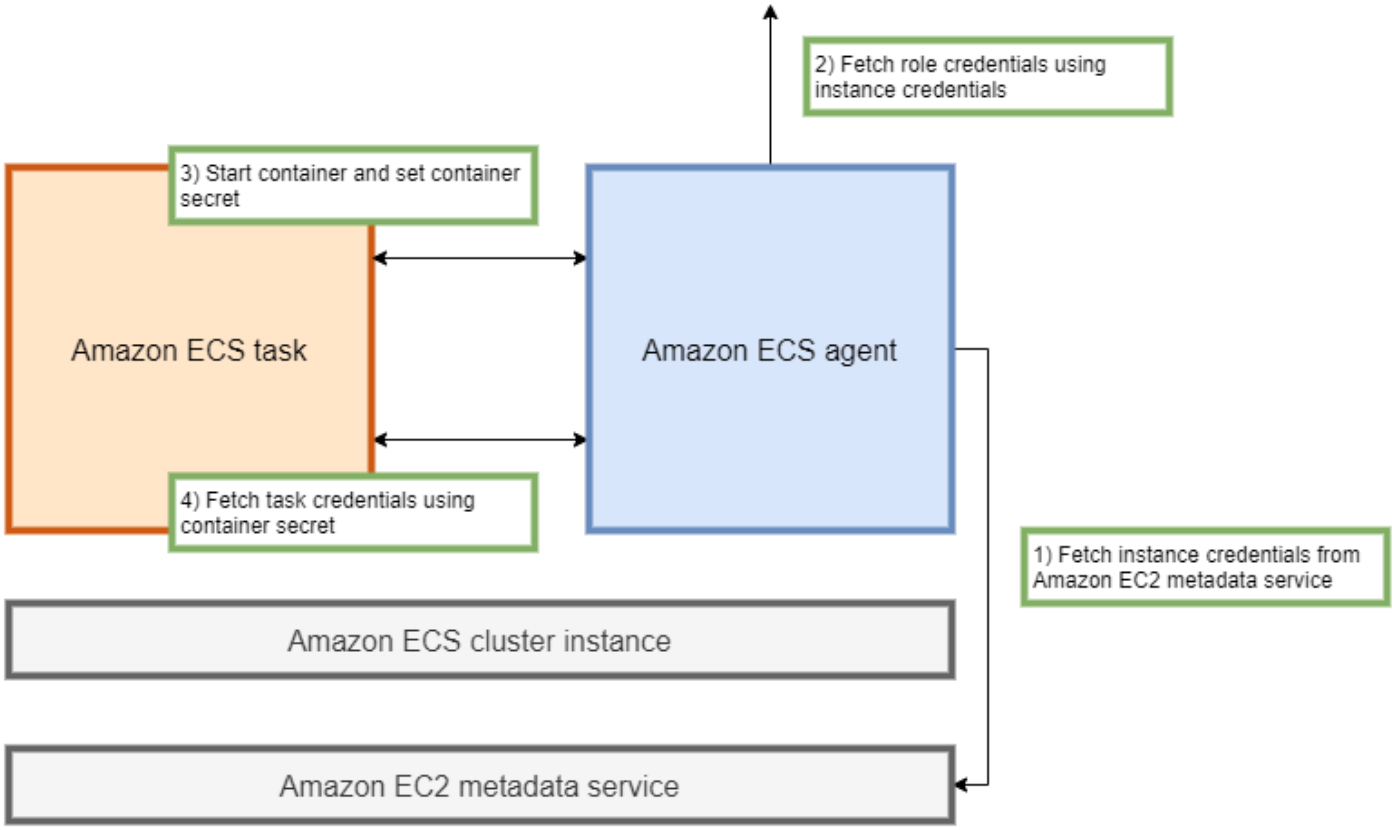
Wir empfehlen, dass Sie eine Aufgabenrolle zuweisen. Ihre Rolle kann von der Rolle der Amazon-EC2-Instance unterschieden werden, auf der sie ausgeführt werden. Die Zuweisung einer Rolle für jede Aufgabe entspricht dem Prinzip des Zugriffs mit der geringsten Berechtigung und ermöglicht eine differenzierte Kontrolle über Aktionen und Ressourcen.

Wenn Sie einer Aufgabe IAM-Rollen zuweisen, müssen Sie die folgende Vertrauensrichtlinie verwenden, damit jede Ihrer Aufgaben eine IAM-Rolle annehmen kann, die sich von der

unterscheidet, die Ihre EC2-Instance verwendet. Auf diese Weise erbt Ihre Aufgabe nicht die Rolle Ihrer EC2-Instance.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Wenn Sie einer Aufgabendefinition eine Aufgabenrolle hinzufügen, erstellt der Amazon-ECS-Container-Agent automatisch ein Token mit einer eindeutigen Anmeldeinformations-ID (z. B. `12345678-90ab-cdef-1234-567890abcdef`) für die Aufgabe. Dieses Token und die Rollen-Anmeldeinformationen werden dann dem internen Cache des Agenten hinzugefügt. Der Agent füllt die Umgebungsvariable `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` im Container mit der URI der Anmeldeinformations-ID auf (z. B. `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).



Sie können die temporären Rollenanmeldedaten manuell aus einem Container abrufen, indem Sie die Umgebungsvariable an die IP-Adresse des Amazon-ECS-Container-Agenten anhängen und den `curl`-Befehl für die resultierende Zeichenfolge ausführen.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

Die erwartete Ausgabe sieht wie folgt aus:

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

Neuere Versionen der AWS SDKs rufen diese Anmeldeinformationen bei AWS API-Aufrufen automatisch aus der `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` Umgebungsvariablen ab.

Die Ausgabe enthält ein Zugriffsschlüsselpaar, das aus einer geheimen Zugriffsschlüssel-ID und einem geheimen Schlüssel besteht, das Ihre Anwendung für den Zugriff auf Ressourcen verwendet. AWS Es enthält auch ein Token, mit dem überprüft AWS wird, ob die Anmeldeinformationen gültig sind. Standardmäßig sind Anmeldeinformationen, die Aufgaben mithilfe von Aufgabenrollen zugewiesen wurden, sechs Stunden lang gültig. Danach werden sie automatisch vom Amazon-ECS-Container-Agenten rotiert.

Aufgabenausführungsrolle

Die Rolle „Aufgabenausführung“ wird verwendet, um dem Amazon ECS-Container-Agenten die Erlaubnis zu erteilen, bestimmte AWS API-Aktionen in Ihrem Namen aufzurufen. Wenn Sie beispielsweise verwenden AWS Fargate, benötigt Fargate eine IAM-Rolle, die es ermöglicht, Bilder aus Amazon ECR abzurufen und Protokolle in Logs zu schreiben. CloudWatch Eine IAM-Rolle ist auch erforderlich, wenn eine Aufgabe auf ein Geheimnis verweist, das in gespeichert ist AWS Secrets Manager, z. B. ein Image-Pull-Secret.

> **Note**
>
> Wenn Sie Images als authentifizierter Benutzer abrufen, ist es weniger wahrscheinlich, dass Sie von den Änderungen an den Pull-Rate-Limits von Docker Hub betroffen sind. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Container-Instances.
> Durch die Verwendung von Amazon ECR und Amazon ECR Public können Sie die von Docker auferlegten Beschränkungen umgehen. Wenn Sie Images von Amazon ECR abrufen, trägt dies auch dazu bei, die Netzwerk-Pull-Zeiten zu verkürzen und Datenübertragungsänderungen zu reduzieren, wenn der Datenverkehr Ihre VPC verlässt.

> **Important**
>
> Wenn Sie Fargate verwenden, müssen Sie sich bei einer privaten Image-Registry mit `repositoryCredentials` authentifizieren. Es ist nicht möglich, die Umgebungsvariablen `ECS_ENGINE_AUTH_TYPE` oder `ECS_ENGINE_AUTH_DATA` für den Amazon-ECS-Container-Agenten festzulegen oder die `ecs.config`-Datei für auf Fargate gehostete Aufgaben zu ändern. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Aufgaben.

Rolle für Container-Instances

Der Amazon-ECS-Container-Agent ist ein Container, der auf jeder Amazon-EC2-Instance in einem Amazon-ECS-Cluster läuft. Sie wird außerhalb von Amazon ECS mithilfe des Befehls `init` initialisiert, der auf dem Betriebssystem verfügbar ist. Folglich können ihr keine Berechtigungen durch eine Aufgabenrolle erteilt werden. Stattdessen müssen die Berechtigungen den Amazon-EC2-Instances zugewiesen werden, auf denen die Agenten ausgeführt werden. Die Aktionsliste in der `AmazonEC2ContainerServiceforEC2Role`-Beispielrichtlinie muss der `ecsInstanceRole` erteilt werden. Wenn Sie dies nicht tun, können Ihre Instances dem Cluster nicht beitreten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

In dieser Richtlinie ermöglichen die `ecr` und `logs` API-Aktionen den Containern, die auf Ihren Instances ausgeführt werden, das Abrufen von Images aus Amazon ECR und das Schreiben von Protokollen in Amazon CloudWatch. Die `ecs`-Aktionen ermöglichen es dem Agenten, Instances

zu registrieren und deren Registrierung aufzuheben und mit der Amazon-ECS-Steuerebene zu kommunizieren. Von diesen ist die Aktion `ecs:CreateCluster` optional.

Service-verknüpfte Rollen

Sie können die serviceverknüpfte Rolle für Amazon ECS verwenden, um dem Amazon-ECS-Service die Berechtigung zu gewähren, andere Service-APIs in Ihrem Namen aufzurufen. Amazon ECS benötigt die Berechtigungen zum Erstellen und Löschen von Netzwerkschnittstellen sowie zum Registrieren und Abmelden von Zielen bei einer Zielgruppe. Außerdem benötigt es die erforderlichen Berechtigungen, um Skalierungsrichtlinien zu erstellen und zu löschen. Diese Berechtigungen werden durch die serviceverknüpfte Rolle erteilt. Diese Rolle wird in Ihrem Namen erstellt, wenn Sie den Service zum ersten Mal verwenden.

> **Note**
>
> Wenn Sie die serviceverknüpfte Rolle versehentlich löschen, können Sie sie neu erstellen. Anweisungen dazu finden Sie unter Die serviceverknüpfte Rolle erstellen.

Empfehlungen für Rollen

Wir empfehlen Ihnen, bei der Einrichtung Ihrer IAM-Rollen und -Richtlinien für Aufgaben wie folgt vorzugehen.

Zugriff auf Amazon-EC2-Metadaten blockieren

Wenn Sie Ihre Aufgaben auf Amazon-EC2 Instances ausführen, empfehlen wir Ihnen dringend, den Zugriff auf Amazon-EC2-Metadaten zu blockieren, um zu verhindern, dass Ihre Container die diesen Instances zugewiesene Rolle erben. Wenn Ihre Anwendungen eine AWS API-Aktion aufrufen müssen, verwenden Sie stattdessen IAM-Rollen für Aufgaben.

Um zu verhindern, dass Aufgaben, die im Bridge-Modus ausgeführt werden, auf Amazon-EC2-Metadaten zugreifen, führen Sie den folgenden Befehl aus oder aktualisieren Sie die Benutzerdaten der Instance. Weitere Anweisungen zum Aktualisieren der Benutzerdaten einer Instance finden Sie in diesem AWS -Support-Artikel. Weitere Informationen zum Bridge-Modus für Aufgabendefinitionen finden Sie unter Netzwerkmodus für Aufgabendefinitionen.

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 192.0.2.0/32 --jump DROP
```

Damit diese Änderung nach einem Neustart bestehen bleibt, führen Sie den folgenden Befehl aus, der für Ihr Amazon Machine Image (AMI) spezifisch ist:

- Amazon Linux 2

```
sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
 iptables
```

- Amazon Linux

```
sudo service iptables save
```

Für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden, setzen Sie die Umgebungsvariable `ECS_AWSVPC_BLOCK_IMDS` in der `/etc/ecs/ecs.config`-Datei auf `true`.

Sie sollten die Variable `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` in der `ecs-agent config`-Datei auf `false` setzen, um zu verhindern, dass die Container, die im `host`-Netzwerk ausgeführt werden, auf die Amazon-EC2-Metadaten zugreifen.

Verwenden Sie den **`awsvpc`** Netzwerkmodus

Verwenden Sie den `awsvpc`-Netzwerkmodus, um den Verkehrsfluss zwischen verschiedenen Aufgaben oder zwischen Ihren Aufgaben und anderen Services, die in Ihrer Amazon VPC ausgeführt werden, einzuschränken. Dies fügt eine zusätzliche Sicherheitsebene hinzu. Der `awsvpc`-Netzwerkmodus bietet Netzwerkisolierung auf Aufgabenebene für Aufgaben, die auf Amazon EC2 ausgeführt werden. Dies ist der Standardmodus aktiviert AWS Fargate. Es ist der einzige Netzwerkmodus, den Sie verwenden können, um Aufgaben eine Sicherheitsgruppe zuzuweisen.

IAM Access Advisor verwenden, um Rollen zu verfeinern

Wir empfehlen, alle Aktionen zu entfernen, die nie oder seit einiger Zeit nicht mehr verwendet wurden. Dadurch wird verhindert, dass unerwünschter Zugriff erfolgt. Überprüfen Sie dazu die von IAM Access Advisor erstellten Ergebnisse und entfernen Sie dann Aktionen, die nie oder in letzter Zeit nicht verwendet wurden. Gehen Sie dazu wie folgt vor.

Führen Sie den folgenden Befehl aus, um einen Bericht mit Informationen über den letzten Zugriff für die betreffende Richtlinie zu erstellen:

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Verwenden Sie die `JobId`, die in der Ausgabe enthalten war, um den folgenden Befehl auszuführen. Nachdem Sie dies getan haben, können Sie die Ergebnisse des Berichts einsehen.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Weitere Informationen finden Sie unter IAM Access Advisor.

Achten Sie AWS CloudTrail auf verdächtige Aktivitäten

Sie können jede verdächtige Aktivität überwachen AWS CloudTrail . Die meisten AWS API-Aufrufe werden AWS CloudTrail als Ereignisse protokolliert. Sie werden von AWS CloudTrail Insights analysiert, und Sie werden über jedes verdächtige Verhalten im Zusammenhang mit `write` API-Aufrufen informiert. Dies kann einen Anstieg des Anrufvolumens beinhalten. Diese Warnmeldungen enthalten Informationen wie den Zeitpunkt, zu dem die ungewöhnliche Aktivität aufgetreten ist, und den wichtigsten Identitäts-ARN, der zu den APIs beigetragen hat.

Sie können Aktionen identifizieren, die von Aufgaben mit einer IAM-Rolle in AWS CloudTrail ausgeführt werden, indem Sie sich die `userIdentity`-Eigenschaft des Ereignisses ansehen. Im folgenden Beispiel besteht das `arn` aus dem Namen der übernommenen Rolle, `s3-write-go-bucket-role`, gefolgt vom Namen der Aufgabe, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Wenn Aufgaben, die eine Rolle übernehmen, auf Amazon-EC2-Container-Instances ausgeführt werden, wird eine Anfrage vom Amazon ECS Container Agent im Audit-Protokoll des Agenten protokolliert, das sich an einer Adresse im `/var/log/ecs/audit.log.YYYY-MM-DD-HH`-Format befindet. Weitere Informationen finden Sie unter Aufgaben-IAM-Rollenprotokoll und Protokollieren von Insights-Ereignissen für Trails.

IAM-Rolle für die Amazon-ECS-Aufgabenausführung

Die Aufgabenausführungsrolle erteilt dem Amazon-ECS-Container- und Fargate-Agenten die Berechtigung, AWS -API-Aufrufe in Ihrem Namen durchzuführen. Die IAM-Rolle für die Aufgabenausführung ist je nach den Anforderungen Ihrer Aufgabe erforderlich. Sie können mehrere Aufgabenausführungsrollen für verschiedene Zwecke und Dienste haben, die Ihrem Konto zugeordnet sind. Informationen zu den IAM-Berechtigungen, die Ihre Anwendung zur Ausführung benötigt, finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.

Im Folgenden sind häufige Anwendungsfälle für eine IAM-Rolle für die Aufgabenausführung aufgeführt:

- Ihre Aufgabe wird auf AWS Fargateoder auf einer externen Instanz gehostet und:
  - ruft ein Container-Image aus einem privaten Amazon ECR-Repository ab.
  - ruft ein Container-Image aus einem privaten Amazon ECR-Repository in einem anderen Konto als dem Konto ab, das die Aufgabe ausführt.
  - sendet CloudWatch Container-Protokolle mithilfe des Protokolltreibers an `awslogs` Logs. Weitere Informationen finden Sie unter Amazon ECS-Protokolle senden an CloudWatch .
- Ihre Aufgaben werden entweder auf Amazon EC2-Instances AWS Fargate oder auf Amazon EC2 EC2-Instances gehostet und:
  - verwendet die private Registrierungsauthentifizierung. Weitere Informationen finden Sie unter Authentifizierungsberechtigungen für private Registrierungen.
  - verwendet Runtime Monitoring.
  - Die Aufgabendefinition verweist mithilfe von Secrets Manager Manager-Geheimnissen oder AWS Systems Manager Parameter Store-Parametern auf sensible Daten. Weitere Informationen finden Sie unter Secrets Manager- oder Systems Manager Manager-Berechtigungen.

> **Note**
>
> Die Aufgabenausführungsrolle wird von Amazon-ECS-Container-Agent Version 1.16.0 und höher unterstützt.

Amazon ECS stellt die verwaltete Richtlinie mit dem Namen AmazonECS bereitTaskExecutionRolePolicy , die die Berechtigungen enthält, die für die oben beschriebenen allgemeinen Anwendungsfälle erforderlich sind. Weitere Informationen finden Sie unter AmazonECS

TaskExecution RolePolicy im AWS Managed Policy Reference Guide. Für spezielle Anwendungsfälle kann es erforderlich sein, Ihrer Rolle zur Aufgabenausführung Inline-Richtlinien hinzuzufügen

Die Amazon ECS-Konsole erstellt eine Rolle zur Aufgabenausführung. Sie können die verwaltete IAM-Richtlinie für Aufgaben manuell anhängen, damit Amazon ECS Berechtigungen für future Funktionen und Verbesserungen hinzufügen kann, sobald diese eingeführt werden. Sie können die Suche in der IAM-Konsole verwenden, um zu suchen `ecsTaskExecutionRole` und festzustellen, ob Ihr Konto bereits über die Rolle „Aufgabenausführung“ verfügt. Weitere Informationen finden Sie unter IAM-Konsolensuche im IAM-Benutzerhandbuch.

Wenn Sie als authentifizierter Benutzer Bilder abrufen, ist es weniger wahrscheinlich, dass Sie von den Änderungen an den Pull-Rate-Limits von Docker Hub betroffen sind. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Container-Instances.

Durch die Verwendung von Amazon ECR und Amazon ECR Public können Sie die von Docker auferlegten Beschränkungen umgehen. Wenn Sie Images von Amazon ECR abrufen, trägt dies auch dazu bei, die Netzwerk-Pull-Zeiten zu verkürzen und Datenübertragungsänderungen zu reduzieren, wenn der Datenverkehr Ihre VPC verlässt.

Wenn Sie Fargate verwenden, müssen Sie sich bei einer privaten Image-Registry mit `repositoryCredentials` authentifizieren. Es ist nicht möglich, die Umgebungsvariablen `ECS_ENGINE_AUTH_TYPE` oder `ECS_ENGINE_AUTH_DATA` für den Amazon-ECS-Container-Agenten festzulegen oder die `ecs.config`-Datei für auf Fargate gehostete Aufgaben zu ändern. Weitere Informationen finden Sie unter Private Registrierungsauthentifizierung für Aufgaben.

Erstellen der -Aufgabenausführungsrolle

Wenn Ihr Konto noch nicht über eine Rolle zur Aufgabenausführung verfügt, gehen Sie wie folgt vor, um die Rolle zu erstellen.

AWS Management Console

So erstellen Sie die Servicerolle für Elastic Container Service (IAM-Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.

4. Wählen Sie für Service oder Anwendungsfall Elastic Container Service und dann den Anwendungsfall Elastic Container Service Task aus.
5. Wählen Sie Weiter aus.
6. Suchen Sie im Abschnitt Berechtigungen hinzufügen nach AmazonECS TaskExecution RolePolicy und wählen Sie dann die Richtlinie aus.
7. Wählen Sie Weiter aus.
8. Geben Sie als Rollenname ecs TaskExecution Role ein.
9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei namens `ecs-tasks-trust-policy.json`, die die Vertrauensrichtlinie enthält, die für die IAM-Rolle verwendet werden soll. Die Datei sollte Folgendes enthalten:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Erstellen Sie eine IAM-Rolle namens `ecsTaskExecutionRole`, die im vorherigen Schritt erstellte Vertrauensrichtlinie verwendet.

```
aws iam create-role \
      --role-name ecsTaskExecutionRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Hängen Sie die AWS verwaltete `AmazonECSTaskExecutionRolePolicy` Richtlinie an die `ecsTaskExecutionRole` Rolle an.

```
aws iam attach-role-policy \
      --role-name ecsTaskExecutionRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

Nachdem Sie die Rolle erstellt haben, fügen Sie der Rolle zusätzliche Berechtigungen für die folgenden Funktionen hinzu.

Funktion	Zusätzliche Berechtigungen
Verwenden Sie Secrets Manager Manager-Anmeldeinformationen, um auf Ihr privates Container-Image-Repository zuzugreifen	Authentifizierungsberechtigungen für private Registrierungen
Übergeben Sie sensible Daten mit Systems Manager oder Secrets Manager	Secrets Manager- oder Systems Manager Manager-Berechtigungen
Lassen Sie Fargate-Aufgaben Amazon ECR-Bilder über Schnittstellenendpunkte abrufen	Fargate-Aufgaben: Abrufen von Amazon ECR-Images über die Berechtigungen von Schnittstellen-Endpunkten
Hosten Sie Konfigurationsdateien in einem Amazon S3 S3-Bucket	Amazon S3 S3-Dateispeicherberechtigungen

Authentifizierungsberechtigungen für private Registrierungen

Um Zugriff auf die Secrets zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur Aufgabendefinitionsrolle hinzufügen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien.

- `secretsmanager:GetSecretValue`
- `kms:Decrypt`: Nur erforderlich, wenn Ihr Schlüssel einen benutzerdefinierten KMS-Schlüssel verwendet und nicht den Standard-Schlüssel. Der Amazon-Ressourcenname (ARN) für Ihren benutzerdefinierten Schlüssel muss als Ressource hinzugefügt werden.

Das folgende Beispiel einer Inline-Richtlinie fügt die Berechtigungen hinzu:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

Secrets Manager- oder Systems Manager Manager-Berechtigungen

Die Erlaubnis, dem Container-Agenten zu erlauben, die erforderlichen AWS Systems Manager oder Secrets Manager Manager-Ressourcen abzurufen. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.

Verwenden von Secrets Manager

Um Zugriff auf die Secrets-Manager-Geheimnisse zu gewähren, die Sie erstellen, müssen Sie die folgenden Berechtigungen manuell zur Aufgabendefinitionsrolle hinzufügen. Informationen zum Verwalten von Berechtigungen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

- `secretsmanager:GetSecretValue` – Erforderlich, wenn Sie auf ein Secrets Manager-Geheimnis verweisen. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
      ]
    }
  ]
}
```

Verwenden von Systems Manager

> **Important**
>
> Für Aufgaben, die den Starttyp EC2 verwenden, müssen Sie die ECS-Agent-Konfigurationsvariable `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` verwenden, um dieses Feature verwenden zu können. Sie können sie während der Erstellung der Container-Instance zur Datei `./etc/ecs/ecs.config` hinzufügen oder sie zu einer vorhandenen Instance hinzufügen und dann den ECS-Agenten neu starten. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Um Zugriff auf die von Ihnen erstellten Systems-Manager-Parameter-Store-Parameter zu erhalten, fügen Sie der Aufgabenausführungsrolle manuell die folgenden Berechtigungen als Inline Richtlinie hinzu. Informationen zum Verwalten von Berechtigungen finden Sie unter Hinzufügen und Entfernen von IAM-Identitätsberechtigungen im IAM-Benutzerhandbuch.

- `ssm:GetParameters` – Erforderlich, wenn in einer Aufgabendefinition auf einen Parameter-Store-Parameter von Systems Manager verwiesen wird. Fügt die Berechtigung zum Abrufen von Systems-Manager-Parametern hinzu.
- `secretsmanager:GetSecretValue` – Erforderlich, wenn Sie direkt auf ein Secrets-Manager-Geheimnis verweisen oder wenn der Parameter Systems Manager Parameter Store in einer Aufgabendefinition auf ein Secrets-Manager-Geheimnis verweist. Fügt die Berechtigung zum Abrufen des Secrets von Secrets Manager hinzu.
- `kms:Decrypt` – Nur erforderlich, wenn Ihr Geheimnis einen kundenverwalteten Schlüssel verwendet und nicht den Standardschlüssel. Der ARN für Ihren benutzerdefinierten Schlüssel sollte

als Ressource hinzugefügt werden. Fügt die Berechtigung zum Entschlüsseln des vom Kunden verwalteten Schlüssels hinzu.

Das folgende Beispiel einer Richtlinie fügt die erforderlichen Berechtigungen hinzu:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Fargate-Aufgaben: Abrufen von Amazon ECR-Images über die Berechtigungen von Schnittstellen-Endpunkten

Beim Starten von Aufgaben, die den Starttyp Fargate verwenden und Images aus Amazon ECR abrufen, wenn Amazon ECR für die Verwendung eines Schnittstellen-VPC-Endpunkts konfiguriert ist, können Sie den Aufgabenzugriff auf eine bestimmte VPC oder einen bestimmten VPC-Endpunkt beschränken. Dazu erstellen Sie eine Aufgabenausführungsrolle für die zu verwendenden Aufgaben, die IAM-Bedingungsschlüssel nutzen.

Verwenden Sie die folgenden globalen IAM-Bedingungsschlüssel zum Einschränken des Zugriffs auf eine bestimmte VPC oder einen bestimmten VPC-Endpunkt. Weitere Informationen finden Sie unter Globale AWS -Bedingungskontextschlüssel.

- `aws:SourceVpc`: Beschränkt den Zugriff auf eine bestimmte VPC.
- `aws:SourceVpce`: Beschränkt den Zugriff auf einen bestimmten VPC-Endpunkt.

Die folgenden Richtlinie einer Aufgabenausführungsrolle stellt ein Beispiel für das Hinzufügen von Bedingungsschlüsseln bereit:

> **Important**
>
> Auf die `ecr:GetAuthorizationToken` API-Aktion können die `aws:sourceVpce` Bedingungsschlüssel `aws:sourceVpc` oder nicht angewendet werden, da der GetAuthorizationToken API-Aufruf über die elastic network interface läuft, die AWS Fargate gehört, und nicht über die elastic network interface der Aufgabe.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpce": "vpce-xxxxxx",
                    "aws:sourceVpc": "vpc-xxxxx"
                }
            }
        }
    ]
}
```

Amazon S3 S3-Dateispeicherberechtigungen

Wenn Sie eine Konfigurationsdatei angeben, die in Amazon S3 gehostet wird, muss Ihre Aufgabenausführungsrolle die `s3:GetObject` Berechtigung für die Konfigurationsdatei und die `s3:GetBucketLocation` Berechtigung für den Amazon S3 S3-Bucket enthalten, in dem sich die Datei befindet. Weitere Informationen finden Sie unter Festlegen von Berechtigungen in einer Richtlinie im Benutzerhandbuch zu Amazon Simple Storage Service.

Die folgende Beispielrichtlinie fügt die erforderlichen Berechtigungen für das Abrufen einer Datei aus Amazon S3 hinzu. Geben Sie den Namen des Amazon S3-Buckets und den Namen der Konfigurationsdatei an.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/folder_name/config_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket"
      ]
    }
  ]
}
```

IAM-Rolle für Amazon ECS-Aufgaben

Ihren Amazon-ECS-Aufgaben kann eine IAM-Rolle zugeordnet sein. Die in der IAM-Rolle gewährten Berechtigungen werden von den Containern übernommen, die in der Aufgabe ausgeführt werden. Diese Rolle ermöglicht es Ihrem Anwendungscode (auf dem Container), andere AWS Dienste zu verwenden. Die Aufgabenrolle ist erforderlich, wenn Ihre Anwendung auf andere AWS Dienste

wie Amazon S3 zugreift. Informationen zu den IAM-Berechtigungen, die Amazon ECS benötigt, um Container-Images abzurufen und die Aufgabe auszuführen, finden Sie unter. IAM-Rolle für die Amazon-ECS-Aufgabenausführung

Im Folgenden sind die Vorteile der Verwendung von Aufgabenrollen aufgeführt:

- Isolierung von Anmeldeinformationen: Ein Container kann nur Anmeldeinformationen für die IAM-Rolle abrufen, die in der Aufgabendefinition definiert ist, zu der er gehört; ein Container hat niemals Zugriff auf Anmeldeinformationen, die für einen anderen Container bestimmt sind, der zu einer anderen Aufgabe gehört.
- Autorisierung: Nicht autorisierte Container können nicht auf IAM-Rollen-Anmeldeinformationen zugreifen, die für andere Aufgaben definiert sind.
- Überwachung: Die Zugriffs- und Ereignisprotokollierung ist bis verfügbar CloudTrail , um eine nachträgliche Prüfung zu gewährleisten. Die Anmeldedaten für Aufgaben haben einen Kontext`taskArn`, der mit der Sitzung verknüpft ist. Aus den CloudTrail Protokollen geht also hervor, welche Aufgabe welche Rolle verwendet.

> **Note**
>
> Wenn Sie eine IAM-Rolle für eine Aufgabe angeben, verwenden die AWS CLI oder andere SDKs in den Containern für diese Aufgabe ausschließlich die von der Aufgabenrolle bereitgestellten AWS Anmeldeinformationen und erben keine IAM-Berechtigungen mehr von der Amazon EC2- oder externen Instance, auf der sie ausgeführt werden.

Die IAM-Rolle für Aufgaben erstellen

Wenn Sie eine IAM-Richtlinie für Ihre Aufgaben erstellen, muss die Richtlinie die Berechtigungen enthalten, die die Container in Ihren Aufgaben annehmen sollen. Sie können eine vorhandene AWS verwaltete Richtlinie verwenden oder Sie können eine benutzerdefinierte Richtlinie von Grund auf neu erstellen, die Ihren spezifischen Anforderungen entspricht. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

> **Important**
>
> Für Amazon-ECS-Aufgaben (für alle Starttypen) empfehlen wir, dass Sie die IAM-Richtlinie und -Rolle für Ihre Aufgaben verwenden. Diese Anmeldeinformationen ermöglichen es Ihrer Aufgabe, AWS API-Anfragen zu stellen, ohne dass Sie aufrufen `sts:AssumeRole`

müssen, um dieselbe Rolle zu übernehmen, die der Aufgabe bereits zugewiesen ist. Wenn Ihr Aufgabe eine Rolle verwendet, um sich selbst anzunehmen, müssen Sie eine Vertrauensrichtlinie erstellen, die ausdrücklich zulässt, dass diese Rolle sich selbst annimmt. Weitere Informationen finden Sie unter Ändern einer Vertrauensrichtlinie für Rollen im IAM-Benutzerhandbuch.

Sobald die IAM-Richtlinie erstellt wurde, können Sie eine IAM-Rolle erstellen, Richtlinie enthält, auf Sie in Ihrer Amazon-ECS-Aufgabendefinition verweisen. Sie können die Rolle mithilfe des Anwendungsfalls Elastic-Container-Service-Aufgabe in der IAM-Konsole erstellen. Anschließend können Sie Ihre spezifische IAM-Richtlinie an die Rolle anhängen, die den Containern in Ihrer Aufgabe die gewünschten Berechtigungen erteilt. Wie Sie dazu vorgehen müssen, ist in den unten stehenden Verfahren beschrieben.

Wenn Sie mehrere Aufgabendefinitionen oder Services haben, die IAM-Berechtigungen benötigen, empfehlen wir, für die jeweiligen Aufgabendefinitionen oder Services eine Rolle mit den zur Ausführung der Aufgaben mindestens benötigten Berechtigungen zu erstellen, um den Zugriff zu minimieren, den Sie für die einzelnen Aufgaben bereitstellen.

Informationen zum Service-Endpunkt für Ihre Region finden Sie unter Service-Endpunkte im Allgemeine Amazon Web Services-Referenz Handbuch.

Die Aufgaben-IAM-Rolle muss über eine Vertrauensrichtlinie verfügen, die den `ecs-tasks.amazonaws.com`-Service angibt. Die `sts:AssumeRole`-Berechtigung erlaubt es Ihren Aufgaben, eine IAM-Rolle anzunehmen, die sich von der unterscheidet, die die Amazon-EC2-Instance verwendet. Auf diese Weise erbt Ihre Aufgabe nicht die der Amazon-EC2-Instance zugeordnete Rolle. Im Folgenden finden Sie ein Beispiel für eine Vertrauensrichtlinie. Ersetzen Sie die Regionskennung und geben Sie die AWS Kontonummer an, die Sie beim Starten von Aufgaben verwenden.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
```

```
            "Action":"sts:AssumeRole",
            "Condition":{
               "ArnLike":{
               "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
               },
               "StringEquals":{
                  "aws:SourceAccount":"111122223333"
               }
            }
         }
      ]
}
```

> **Important**
>
> Es wird empfohlen, bei der Erstellung Ihrer Aufgaben-IAM-Rolle die `aws:SourceArn` Bedingungsschlüssel oder entweder in der Vertrauensstellung `aws:SourceAccount` oder in der IAM-Richtlinie, die der Rolle zugeordnet ist, zu verwenden, um die Berechtigungen weiter einzuschränken und so das verwirrende Sicherheitsproblem des stellvertretenden Mitarbeiters zu vermeiden. Verwendung des `aws:SourceArn` Bedingungsschlüssels zur Angabe eines bestimmten Clusters wird derzeit nicht unterstützt. Sie sollten den Platzhalter verwenden, um alle Cluster anzugeben. Weitere Informationen über das Problem mit dem verwirrten Stellvertreter und darüber, wie Sie Ihr AWS Konto schützen können, finden Sie im IAM-Benutzerhandbuch unter Das Problem mit dem verwirrten Stellvertreter.

In den folgenden Verfahren wird anhand einer Beispielrichtlinie beschrieben, wie Sie eine Richtlinie zum Abrufen von Objekten aus Amazon S3 erstellen. Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Werte.

AWS Management Console

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.

3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ],
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:region:123456789012:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            }
         }
      }
   ]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.

8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Werte.

1. Erstellen Sie eine Datei mit dem Namen `s3-policy.json` und folgendem Inhalt.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ],
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:region:123456789012:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            }
         }
      }
   ]
}
```

2. Verwenden Sie den folgenden Befehl, um die IAM-Richtlinie mithilfe der JSON-Richtliniendokumentdatei zu erstellen.

```
aws iam create-policy \
      --policy-name taskRolePolicy \
      --policy-document file://s3-policy.json
```

In den folgenden Verfahren wird beschrieben, wie Sie eine Task-IAM-Rolle erstellen, indem Sie eine von Ihnen erstellte IAM-Richtlinie anhängen.

AWS Management Console

So erstellen Sie die Servicerolle für Elastic Container Service (IAM-Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
4. Wählen Sie für Service oder Anwendungsfall Elastic Container Service und dann den Anwendungsfall Elastic Container Service Task aus.
5. Wählen Sie Weiter aus.
6. Suchen Sie unter Berechtigungen hinzufügen nach der Richtlinie, die Sie erstellt haben, und wählen Sie sie aus.
7. Wählen Sie Weiter aus.
8. Geben Sie unter Role name (Rollenname) einen Namen für Ihre Rolle ein. Geben Sie für dieses Beispiel `AmazonECSTaskS3BucketRole` ein, um die Rolle zu benennen.
9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

AWS CLI

Ersetzen Sie alle *`Benutzereingaben`* durch Ihre eigenen Werte.

1. Erstellen Sie eine Datei mit dem Name`ecs-tasks-trust-policy.json`, die die Vertrauensrichtlinie enthält, die für die IAM-Rolle der Aufgabe verwendet werden soll. Die Datei sollte Folgendes enthalten. Ersetzen Sie die Regionskennung und geben Sie die AWS Kontonummer an, die Sie beim Starten von Aufgaben verwenden.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
```

```
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
   ]
}
```

2. Erstellen Sie eine IAM-Rolle namens `ecsTaskRole`, die im vorherigen Schritt erstellte Vertrauensrichtlinie verwendet.

```
aws iam create-role \
      --role-name ecsTaskRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Rufen Sie den ARN der IAM-Richtlinie ab, die Sie mit dem folgenden Befehl erstellt haben. Ersetzen Sie die *`Aufgabe RolePolicy`* durch den Namen der Richtlinie, die Sie erstellt haben.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`taskRolePolicy`].Arn'
```

4. Hängen Sie die von Ihnen erstellte IAM-Richtlinie an die `ecsTaskRole` Rolle an. Ersetzen Sie das `policy-arn` durch den ARN der Richtlinie, die Sie erstellt haben.

```
aws iam attach-role-policy \
      --role-name ecsTaskRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/taskRolePolicy
```

Nachdem Sie die Rolle erstellt haben, fügen Sie der Rolle zusätzliche Berechtigungen für die folgenden Funktionen hinzu.

Funktion	Zusätzliche Berechtigungen
Verwenden Sie ECS Exec	ECS Exec-Berechtigungen
Verwenden Sie EC2-Instances (Windows und Linux)	Zusätzliche Konfiguration Amazon EC2 EC2-Instances
Verwenden Sie externe Instanzen	Zusätzliche Konfiguration der externen Instanz
Verwenden Sie Windows EC2-Instanzen	Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance

ECS Exec-Berechtigungen

Für die ECS Exec-Funktion ist eine Task-IAM-Rolle erforderlich, um Containern die für die Kommunikation zwischen dem verwalteten SSM-Agenten (`execute-command`Agenten) und dem SSM-Dienst erforderlichen Berechtigungen zu gewähren. Sie sollten einer Aufgaben-IAM-Rolle die folgenden Berechtigungen hinzufügen und die Aufgaben-IAM-Rolle in Ihre Aufgabendefinition aufnehmen. Weitere Informationen finden Sie unter Hinzufügen und Entfernen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Verwenden Sie die folgende Richtlinie für Ihre Aufgaben-IAM-Rolle, um die erforderlichen SSM-Berechtigungen hinzuzufügen.

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
       ],
      "Resource": "*"
      }
   ]
}
```

Zusätzliche Konfiguration Amazon EC2 EC2-Instances

Wir empfehlen Ihnen, die Berechtigungen in Ihrer Container-Instance-Rolle auf die minimale Liste von Berechtigungen, die in der verwalteten IAM-Richtlinie `AmazonEC2ContainerServiceforEC2Role` verwendet werden, zu begrenzen.

Ihre Amazon EC2 EC2-Instances benötigen mindestens die Version `1.11.0` des Container-Agenten, um die Aufgabenrolle verwenden zu können. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten. Wenn Sie ein Amazon ECS-optimiertes AMI verwenden, benötigt Ihre Instance mindestens einen Teil `1.11.0-1` des `ecs-init` Pakets. Wenn Ihre Instances das neueste Amazon-ECS-optimierte AMI verwenden, enthalten sie die erforderlichen Versionen des Container-Agenten und `ecs-init`. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Wenn Sie das Amazon ECS-optimierte AMI nicht für Ihre Container-Instances verwenden, fügen Sie Ihrem docker run Befehl die `--net=host` Option hinzu, mit der der Agent gestartet wird, und die folgenden Agentenkonfigurationsvariablen für Ihre gewünschte Konfiguration (weitere Informationen finden Sie unterKonfiguration des Amazon-ECS-Container-Agenten):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Verwendet IAM-Rollen für Aufgaben für Container mit den Netzwerkmodi `bridge` und `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Verwendet IAM-Rollen für Aufgaben für Container mit dem `host`-Netzwerkmodus. Diese Variable wird nur von den Agenten-Versionen 1.12.0 oder höher unterstützt.

Ein Beispiel für einen Ausführungsbefehl finden Sie unter Manuelles Aktualisieren des Amazon-ECS-Container-Agenten (für Nicht-Amazon-ECS-optimierte AMIs). Sie müssen außerdem die folgenden Netzwerkbefehle auf Ihrer Container-Instance einrichten, damit die Container in Ihren Aufgaben ihre AWS Anmeldeinformationen abrufen können:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Sie müssen diese iptables-Regeln auf Ihrer Container-Instance speichern, damit sie einen Neustart überstehen. Sie können mithilfe der Befehle iptables-save und iptables-restore Ihre iptables-Regeln speichern und während des Boot-Prozesses wiederherstellen. Weitere Informationen finden Sie in der Dokumentation zu Ihrem entsprechenden Betriebssystem.

Um zu verhindern, dass Container, die von Aufgaben mit dem `awsvpc`-Netzwerkmodus ausgeführt werden, auf die Anmeldeinformationen zugreifen, die dem Amazon-EC2-Instance-Profil bereitgestellt werden, während die von der Aufgabenrolle bereitgestellten Berechtigungen weiterhin erlaubt werden, setzen Sie die Konfigurationsvariable für den `ECS_AWSVPC_BLOCK_IMDS`-Agenten in der Agenten-Konfigurationsdatei auf `true` und starten Sie den Agenten neu. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten.

Verhindern Sie, dass Container, die von Aufgaben mit dem `bridge`-Netzwerkmodus ausgeführt werden, auf die Anmeldeinformationen zugreifen, die dem Amazon-EC2-Instance-Profil bereitgestellt werden, während die von der Aufgabenrolle bereitgestellten Berechtigungen weiterhin erlaubt werden, führen Sie den folgenden iptables-Befehl auf Ihren Amazon-EC2-Instances aus. Dieser Befehl wirkt sich nicht auf Container in Aufgaben aus, die den `host`- oder `awsvpc`-Netzwerkmodus verwenden. Weitere Informationen finden Sie unter Netzwerkmodus.

- ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER-USER 1 --in-interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Sie müssen diese iptables-Regel auf Ihrer Amazon-EC2-Instance speichern, damit sie einen Neustart übersteht. Wenn Sie das Amazon-ECS-optimierte AMI verwenden, können Sie den folgenden Befehl verwenden. Informationen über andere Betriebssysteme finden Sie in der Dokumentation des jeweiligen Betriebssystems.

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now iptables
  ```

Zusätzliche Konfiguration der externen Instanz

Ihre externen Instances benötigen mindestens die Version `1.11.0` des Container-Agenten, um Task-IAM-Rollen verwenden zu können. Wir empfehlen jedoch, die neueste Container-Agent-Version zu verwenden. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten. Wenn Sie das Amazon-ECS-optimierte AMI verwenden, benötigt Ihre Instance mindestens `1.11.0-1` des `ecs-`
  ```

`init`-Pakets. Wenn Ihre Instances das neueste Amazon-ECS-optimierte AMI verwenden, enthalten sie die erforderlichen Versionen des Container-Agenten und `ecs-init`. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.

Wenn Sie das Amazon ECS-optimierte AMI nicht für Ihre Container-Instances verwenden, fügen Sie Ihrem docker run Befehl die `--net=host` Option hinzu, mit der der Agent gestartet wird, und die folgenden Agentenkonfigurationsvariablen für Ihre gewünschte Konfiguration (weitere Informationen finden Sie unterKonfiguration des Amazon-ECS-Container-Agenten):

`ECS_ENABLE_TASK_IAM_ROLE=true`

Verwendet IAM-Rollen für Aufgaben für Container mit den Netzwerkmodi `bridge` und `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Verwendet IAM-Rollen für Aufgaben für Container mit dem `host`-Netzwerkmodus. Diese Variable wird nur von den Agenten-Versionen 1.12.0 oder höher unterstützt.

Ein Beispiel für einen Ausführungsbefehl finden Sie unter Manuelles Aktualisieren des Amazon-ECS-Container-Agenten (für Nicht-Amazon-ECS-optimierte AMIs). Sie müssen außerdem die folgenden Netzwerkbefehle auf Ihrer Container-Instance einrichten, damit die Container in Ihren Aufgaben ihre AWS Anmeldeinformationen abrufen können:

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Sie müssen diese iptables-Regeln auf Ihrer Container-Instance speichern, damit sie einen Neustart überstehen. Sie können mithilfe der Befehle iptables-save und iptables-restore Ihre iptables-Regeln speichern und während des Boot-Prozesses wiederherstellen. Weitere Informationen finden Sie in der Dokumentation zu Ihrem entsprechenden Betriebssystem.

Zusätzliche Konfiguration der Amazon EC2 EC2-Windows-Instance

> **Important**
>
> Dies gilt nur für Windows-Container auf EC2, die Aufgabenrollen verwenden.

Die Aufgabenrolle mit Windows-Funktionen erfordert eine zusätzliche Konfiguration auf EC2.

- Wenn Sie Container-Instances starten, müssen Sie die Option `-EnableTaskIAMRole` im Benutzerdatenskript der Container-Instances festlegen. `EnableTaskIAMRole` aktiviert die Task-IAM-Rollenfunktion für die Aufgaben. Zum Beispiel:

  ```
  <powershell>
  Import-Module ECSTools
  Initialize-ECSAgent -Cluster 'windows' -EnableTaskIAMRole
  </powershell>
  ```

- Sie müssen einen Bootstrap für Ihren Container mit den Netzwerkbefehlen ausführen, die unter Amazon ECS-Container-Bootstrap-Skript angegeben sind.
- Sie müssen eine IAM-Rolle und -Richtlinie für Ihre Aufgaben erstellen. Weitere Informationen finden Sie unter Die IAM-Rolle für Aufgaben erstellen.
- Die IAM-Rollen für den Anbieter der Anmeldeinformationen für die Aufgabe verwenden Port 80 auf der Container-Instance. Daher können Ihre Container nicht Port 80 für den Host-Port in Port-Mappings verwenden, wenn Sie IAM-Rollen für Aufgaben auf Ihrer Container-Instance konfigurieren. Um Ihre Container auf Port 80 bereitzustellen, empfehlen wir, dass Sie dafür einen Container mit Lastausgleich konfigurieren. Sie können Port 80 auf dem Load Balancer verwenden. Auf diese Weise kann der Datenverkehr an einen anderen Host-Port auf Ihren Container-Instances umgeleitet werden. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.
- Wenn Ihre Windows-Instance neu gestartet wird, müssen Sie die Proxy-Schnittstelle löschen und den Amazon-ECS-Container-Agenten erneut initialisieren, um den Anmeldeinformationsproxy wieder zu starten.

Amazon ECS-Container-Bootstrap-Skript

Bevor Container zum Abrufen von Anmeldeinformationen auf den Proxy für Anmeldeinformationen auf der Container-Instance zugreifen können, muss für den Container ein Bootstrap mit den erforderlichen Netzwerkbefehlen ausgeführt werden. Das folgende Code-Beispielskript sollte auf Ihren Containern bei deren Start ausgeführt werden.

> **Note**
>
> Sie müssen dieses Skript nicht ausführen, wenn Sie `awsvpc`-Netzwerkmodus unter Windows verwenden.

Wenn Sie Windows-Container mit Powershell ausführen, verwenden Sie das folgende Skript:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

$gateway = (Get-NetRoute | Where { $_.DestinationPrefix -eq '0.0.0.0/0' } | Sort-Object
 RouteMetric | Select NextHop).NextHop
$ifIndex = (Get-NetAdapter -InterfaceDescription "Hyper-V Virtual Ethernet*" | Sort-
Object | Select ifIndex).ifIndex
New-NetRoute -DestinationPrefix 169.254.170.2/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # credentials API
New-NetRoute -DestinationPrefix 169.254.169.254/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # metadata API
```

Wenn Sie Windows-Container ausführen, die nur die Befehls-Shell haben, verwenden Sie das folgende Skript:

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
```

```
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

for /f "tokens=1" %i  in ('netsh interface ipv4 show interfaces ^| findstr /x /r
 ".*vEthernet.*"') do set interface=%i
for /f "tokens=3" %i  in ('netsh interface ipv4 show addresses %interface% ^| findstr /
x /r ".*Default.Gateway.*"') do set gateway=%i
netsh interface ipv4 add route prefix=169.254.170.2/32 interface="%interface%"
 nexthop="%gateway%" store=active # credentials API
netsh interface ipv4 add route prefix=169.254.169.254/32 interface="%interface%"
 nexthop="%gateway%" store=active # metadata API
```

IAM-Rolle für Amazon-ECS-Container-Instance

Amazon-ECS-Container-Instances, einschließlich Amazon EC2 und externer Instances, führen den Amazon-ECS-Container-Agent aus und benötigen daher eine IAM-Rolle, damit der Service weiß, dass der Agent zu Ihnen gehört. Bevor Sie Container-Instances starten und sie in einem Cluster registrieren, müssen Sie eine IAM-Rolle erstellen, die ihre Container-Instances verwenden können. Die Rolle wird in dem Konto erstellt, mit dem Sie sich bei der Konsole anmelden oder die AWS CLI Befehle ausführen.

> **Important**
>
> Wenn Sie externe Instances in Ihrem Cluster registrieren, erfordert die von Ihnen verwendete IAM-Rolle ebenfalls die Systems Manager Berechtigungen. Weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere.

Amazon ECS stellt die von `AmazonEC2ContainerServiceforEC2Role` verwaltete IAM-Richtlinie, die die Berechtigungen enthält, die für die Verwendung des vollständigen Amazon-ECS-Featuresatzes erforderlich sind, bereit. Diese verwaltete Richtlinie kann einer IAM-Rolle zugeordnet und Ihren Container-Instances zugeordnet werden. Alternativ können Sie die verwaltete Richtlinie als Leitfaden verwenden, wenn Sie eine benutzerdefinierte Richtlinie verwenden möchten. Die Container-Instance-Rolle stellt die erforderlichen Berechtigungen bereit, damit der Amazon ECS-Container-Agent und der Docker-Daemon AWS APIs in Ihrem Namen aufrufen können. Für weitere Informationen über die verwaltete Richtlinie siehe Amazon EC2 EC2-Rolle ContainerServicefor.

Amazon ECS unterstützt das Starten von Container-Instances mit erhöhter ENI-Dichte mit unterstützten Amazon-EC2-Instance-Typen. Wenn Sie diese Funktion verwenden, empfehlen

wir Ihnen, zwei Container-Instance-Rollen zu erstellen. Aktivieren Sie die `awsvpcTrunking` Kontoeinstellung für eine Rolle und verwenden Sie diese Rolle für Aufgaben, die ENI-Trunking erfordern. Informationen zur `awsvpcTrunking` Kontoeinstellung finden Sie unterGreifen Sie mit Kontoeinstellungen auf Amazon ECS-Funktionen zu.

Erstellen Sie die Container-Instance-Rolle

> **Important**
>
> Wenn Sie externe Instances in Ihrem Cluster registrieren, finden Sie weitere Informationen unter IAM-Rolle in Amazon ECS Anywhere.

Sie können jedoch die verwaltete IAM-Richtlinie für Container-Instances manuell erstellen und anfügen, damit Amazon ECS die Berechtigungen für zukünftige Features und Erweiterungen bei deren Einführung hinzufügen kann. Gehen Sie wie folgt vor, um die verwaltete IAM-Richtlinie bei Bedarf anzuhängen.

AWS Management Console

So erstellen Sie die Servicerolle für Elastic Container Service (IAM-Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
4. Wählen Sie für Service oder Anwendungsfall Elastic Container Service und dann den Anwendungsfall EC2-Rolle für Elastic Container Service aus.
5. Wählen Sie Weiter aus.
6. Vergewissern Sie sich im Abschnitt Permissions Policies, dass die AmazonEC2 ContainerServicefor EC2Role-Richtlinie ausgewählt ist.

   > **Important**
   >
   > Die verwaltete AmazonEC2 ContainerServicefor EC2Role-Richtlinie sollte an die IAM-Rolle der Container-Instance angehängt werden. Andernfalls erhalten Sie bei der

Verwendung von zum Erstellen von Clustern eine Fehlermeldung. AWS Management Console

7. Wählen Sie Weiter aus.
8. Geben Sie als Rollenname ecs ein InstanceRole
9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Werte.

1. Erstellen Sie eine Datei mit dem Namen `instance-role-trust-policy.json` und den folgenden Inhalten.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "Service": "ec2.amazonaws.com"},
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Verwenden Sie den folgenden Befehl, um die Instance-IAM-Rolle mithilfe des Trust Policy-Dokuments zu erstellen.

```
aws iam create-role \
    --role-name ecsInstanceRole \
    --assume-role-policy-document file://instance-role-trust-policy.json
```

3. Erstellen Sie ein Instance-Profil namens „`ecsInstanceRole-profile`“, indem Sie den Befehl create-instance-profile verwenden.

```
aws iam create-instance-profile --instance-profile-name ecsInstanceRole-profile
```

Beispielantwort

```
{
```

```
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
        "Roles": [],
        "CreateDate": "2022-04-12T23:53:34.093Z",
        "InstanceProfileName": "ecsInstanceRole-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole-
profile"
    }
}
```

4. Fügen Sie dem Instance-Profil *ecsInstanceRole* die IAM-Rolle *ecsInstanceRole-profile* hinzu.

```
aws iam add-role-to-instance-profile \
    --instance-profile-name ecsInstanceRole-profile \
    --role-name ecsInstanceRole
```

5. Hängen Sie die `AmazonEC2ContainerServiceRoleForEC2Role` verwaltete Richtlinie mit dem folgenden Befehl an die Rolle an.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Nachdem Sie die Rolle erstellt haben, fügen Sie der Rolle zusätzliche Berechtigungen für die folgenden Funktionen hinzu.

Funktion	Zusätzliche Berechtigungen
Amazon ECR hat das Container-Image	Amazon-ECR-Berechtigungen
Lassen Sie Container-Instances von CloudWatch Logs überwachen	Überwachung der Berechtigungen für Container-Instances
Hosten Sie Konfigurationsdateien in einem Amazon S3 S3-Bucket	Amazon S3 S3-Lesezugriff

Amazon-ECR-Berechtigungen

Die Amazon ECS-Container-Instance-Rolle, die Sie mit Ihren Container-Instances verwenden, muss über die folgenden IAM-Richtlinienberechtigungen für Amazon ECR verfügen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die verwaltete Richtlinie `AmazonEC2ContainerServiceforEC2Role` für die Container-Instances nutzen, weist die Rolle die erforderlichen Berechtigungen auf. Überprüfen Sie, ob Ihre Rolle Amazon ECR unterstützt in IAM-Rolle für Amazon ECS-Container-Instance im Amazon Elastic Container Service-Entwicklerhandbuch.

Amazon S3 S3-Lesezugriff

Das Speichern von Konfigurationsinformationen in einem privaten Amazon S3-Bucket und das Erteilen der schreibgeschützten Zugriffsberechtigung Ihrer IAM-Rolle der Container-Instance ist eine sichere und bequeme Art, die Konfiguration der Container-Instance zur Startzeit zu ermöglichen. Sie können eine Kopie Ihrer `ecs.config` Datei in einem privaten Bucket speichern, Amazon EC2 EC2-Benutzerdaten zur Installation verwenden AWS CLI und dann Ihre Konfigurationsinformationen zum `/etc/ecs/ecs.config` Start der Instance kopieren.

Weitere Informationen dazu, wie Sie eine Datei `ecs.config` erstellen, sie in Amazon S3 speichern und Instances mit dieser Konfiguration starten, finden Sie unter Speichern der Amazon ECS-Container-Instance-Konfiguration in Amazon S3.

Sie können den folgenden AWS CLI Befehl verwenden, um Amazon S3 nur Lesezugriff für Ihre Container-Instance-Rolle zu gewähren. Ersetzen Sie *ecs InstanceRole* durch den Namen der Rolle, die Sie erstellt haben.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws::iam::aws:policy/AmazonS3ReadOnlyAccess
```

Sie können Ihrer Rolle auch die IAM-Konsole verwenden, um Amazon S3 S3-Lesezugriff (`AmazonS3ReadOnlyAccess`) hinzuzufügen. Weitere Informationen finden Sie im Benutzerhandbuch unter Ändern einer Rollenberechtigungsrichtlinie (Konsole).AWS Identity and Access Management

Überwachung der Berechtigungen für Container-Instances

Bevor Ihre Container-Instances Protokolldaten an CloudWatch Logs senden können, müssen Sie eine IAM-Richtlinie erstellen, die es Ihren Container-Instances ermöglicht, die CloudWatch Logs-APIs zu verwenden, und dann müssen Sie diese Richtlinie anhängen. `ecsInstanceRole`

AWS Management Console

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Nachdem Sie die Richtlinie erstellt haben, fügen Sie die Richtlinie der Container-Instance-Rolle hinzu. Informationen zum Anhängen der Richtlinie an die Rolle finden Sie unter Ändern einer Rollenberechtigungsrichtlinie (Konsole) im AWS Identity and Access Management Benutzerhandbuch.

AWS CLI

1. Erstellen Sie eine Datei mit dem Namen `instance-cw-logs.json` und folgendem Inhalt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

2. Verwenden Sie den folgenden Befehl, um die IAM-Richtlinie mithilfe der JSON-Richtliniendokumentdatei zu erstellen.

```
aws iam create-policy \
      --policy-name cwlogspolicy \
      --policy-document file://instance-cw-logs.json
```

3. Rufen Sie den ARN der IAM-Richtlinie ab, die Sie mit dem folgenden Befehl erstellt haben. Ersetzen Sie *cwlogspolicy* durch den Namen der Richtlinie, die Sie erstellt haben.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogsppolicy`].Arn'
```

4. Verwenden Sie den folgenden Befehl, um die Richtlinie mithilfe des Richtlinien-ARN an die IAM-Rolle der Container-Instance anzuhängen.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cwlogspolicy
```

IAM-Rolle in Amazon ECS Anywhere

Wenn Sie einen lokalen Server oder eine virtuelle Maschine (VM) in Ihrem Cluster registrieren, benötigt der Server oder die VM eine IAM-Rolle, um mit AWS APIs zu kommunizieren. Sie müssen diese IAM-Rolle nur einmal für jedes Konto erstellen. AWS Diese IAM-Rolle muss jedoch jedem Server oder VM zugeordnet werden, den/die Sie bei einem Cluster registrieren. Diese Rolle ist `ECSAnywhereRole`. Sie können diese Rolle manuell erstellen. Alternativ kann Amazon ECS die Rolle in Ihrem Namen erstellen, wenn Sie eine externe Instance in AWS Management Console registrieren. Sie können die Suche in der IAM-Konsole verwenden, um nach der Rolle zu suchen

`ecsAnywhereRole` und festzustellen, ob Ihr Konto bereits über diese Rolle verfügt. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Suche in der IAM-Konsole.

AWS stellt zwei verwaltete IAM-Richtlinien bereit, die bei der Erstellung der ECS Anywhere-IAM-Rolle verwendet werden können: die Richtlinien und. `AmazonSSMManagedInstanceCore` `AmazonEC2ContainerServiceforEC2Role` Die `AmazonEC2ContainerServiceforEC2Role`-Richtlinie enthält Berechtigungen, die wahrscheinlich mehr Zugriff bieten, als Sie benötigen. Daher empfiehlt es sich, je nach Anwendungsfall eine benutzerdefinierte Richtlinie zu erstellen, die nur die Berechtigungen dieser Richtlinie hinzufügt, die Sie in dieser Richtlinie benötigen. Weitere Informationen finden Sie unter IAM-Rolle für Amazon-ECS-Container-Instance.

Die IAM-Rolle für die Aufgabenausführung, die dem Amazon-ECS-Containeragenten die Berechtigung erteilt, AWS -API-Aufrufe in Ihrem Namen durchzuführen. Wenn eine IAM-Rolle für die Aufgabenausführung verwendet wird, muss sie in der Aufgabendefinition angegeben werden. Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

Die Aufgabenausführungsrolle ist erforderlich, wenn eine der folgenden Bedingungen zutrifft:

- Sie senden Container-Logs mithilfe des CloudWatch Log-Treibers an `awslogs` Logs.
- Ihre Aufgabendefinition gibt ein Container-Image an, das in einem privaten Amazon ECR-Repository gehostet wird. Wenn die `ECSAnywhereRole` Rolle, die Ihrer externen Instance zugeordnet ist, jedoch auch die zum Abrufen von Bildern aus Amazon ECR erforderlichen Berechtigungen umfasst, muss Ihre Aufgabenausführungsrolle diese nicht enthalten.

Die Amazon ECS Anywhere Anywhere-Rolle erstellen

Ersetzen Sie alle *`Benutzereingaben`* durch Ihre eigenen Informationen.

1. Erstellen Sie eine lokale Datei `ssm-trust-policy.json` mit dem Namen der folgenden Vertrauensrichtlinie.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": [
      "ssm.amazonaws.com"
    ]},
    "Action": "sts:AssumeRole"
  }
```

```
}
```

2. Erstellen Sie die Rolle und fügen Sie die Vertrauensrichtlinie mit dem folgenden AWS CLI Befehl hinzu.

```
aws iam create-role --role-name ecsAnywhereRole --assume-role-policy-document
 file://ssm-trust-policy.json
```

3. Fügen Sie die AWS verwalteten Richtlinien mit dem folgenden Befehl an.

```
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
```

Sie können die Rolle auch mithilfe des Workflows für benutzerdefinierte IAM-Vertrauensrichtlinien erstellen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole).

IAM-Rolle für die Amazon ECS-Infrastruktur

Eine Amazon ECS-Infrastruktur-IAM-Rolle ermöglicht Amazon ECS, die Infrastrukturressourcen in Ihren Clustern in Ihrem Namen zu verwalten. Sie wird verwendet, wenn:

- Sie möchten Amazon EBS-Volumes an Ihre Amazon ECS-Aufgaben vom Starttyp Fargate oder EC2 anhängen. Die Infrastrukturrolle ermöglicht es Amazon ECS, Amazon EBS-Volumes für Ihre Aufgaben zu verwalten.
- Sie möchten Transport Layer Security (TLS) verwenden, um den Verkehr zwischen Ihren Amazon ECS Service Connect-Services zu verschlüsseln.

Wenn Amazon ECS diese Rolle übernimmt, um in Ihrem Namen Maßnahmen zu ergreifen, werden die Ereignisse in sichtbar sein AWS CloudTrail. Wenn Amazon ECS die Rolle zur Verwaltung von Amazon EBS-Volumes verwendet, die Ihren Aufgaben zugeordnet sind, `roleSessionNamefolefole` wird `ECSTaskVolumesForEBS` das CloudTrail Protokoll gespeichert. Wenn die Rolle zur Verschlüsselung des Datenverkehrs zwischen Ihren Amazon ECS Service Connect-Services verwendet wird, `roleSessionName` wird `ECSServiceConnectForTLS` das CloudTrail Protokoll gespeichert. Sie können diesen Namen verwenden, um Ereignisse in der CloudTrail Konsole zu suchen, indem Sie nach dem Benutzernamen filtern.

Amazon ECS bietet verwaltete Richtlinien, die die für Volume Attachment und TLS erforderlichen Berechtigungen enthalten. Weitere Informationen finden Sie unter AmazonECS InfrastructureRole PolicyFor Volumes und AmazonECS InfrastructureRole PolicyFor ServiceConnect TransportLayer Security im AWS Managed Policy Reference Guide.

Die Amazon ECS-Infrastrukturrolle erstellen

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei namens `ecs-infrastructure-trust-policy.json`, die die Vertrauensrichtlinie enthält, die für die IAM-Rolle verwendet werden soll. Die Datei sollte Folgendes enthalten:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Verwenden Sie den folgenden AWS CLI Befehl, um eine Rolle zu erstellen, die `ecsInfrastructureRole` mithilfe der Vertrauensrichtlinie benannt ist, die Sie im vorherigen Schritt erstellt haben.

```
aws iam create-role \
      --role-name ecsInfrastructureRole \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Hängen Sie je nach Anwendungsfall die AWS verwaltete Rolle `AmazonECSInfrastructureRolePolicyForVolumes` oder die `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` Richtlinie an die `ecsInfrastructureRole` Rolle an.

```
aws iam attach-role-policy \
```

```
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForVolumes
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity
```

Sie können die Rolle auch mithilfe des Workflows für benutzerdefinierte Vertrauensrichtlinien der IAM-Konsole erstellen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole).

Important

Wenn die ECS-Infrastrukturrolle von Amazon ECS zur Verwaltung von Amazon EBS-Volumes verwendet wird, die Ihren Aufgaben zugeordnet sind, stellen Sie Folgendes sicher, bevor Sie Aufgaben beenden, die Amazon EBS-Volumes verwenden.

- Die Rolle wird nicht gelöscht.
- Die Vertrauensrichtlinie für die Rolle wurde nicht geändert, um den Amazon ECS-Zugriff zu entfernen (`ecs.amazonaws.com`).
- Die verwaltete Richtlinie wurde `AmazonECSInfrastructureRolePolicyForVolumes` nicht entfernt. Wenn Sie die Berechtigungen der Rolle ändern müssen, behalten Sie mindestens, bei `ec2:DetachVolumeec2:DeleteVolume`, und `ec2:DescribeVolumes` für das Löschen von Volumes.

Das Löschen oder Ändern der Rolle vor dem Beenden von Aufgaben mit angehängten Amazon EBS-Volumes führt dazu, dass die Aufgaben hängen bleiben `DEPROVISIONING` und die zugehörigen Amazon EBS-Volumes nicht gelöscht werden können. Amazon ECS versucht in regelmäßigen Abständen automatisch erneut, die Aufgabe zu beenden und das Volume zu löschen, bis die erforderlichen Berechtigungen wiederhergestellt sind. Mithilfe der DescribeTasksAPI können Sie den Status des Volumen-Anhangs einer Aufgabe und den zugehörigen Statusgrund einsehen.

Nachdem Sie die Datei erstellt haben, müssen Sie Ihrem Benutzer die Erlaubnis erteilen, die Rolle an Amazon ECS weiterzugeben.

Erlaubnis zur Weitergabe der Infrastrukturrolle an Amazon ECS

Um eine ECS-Infrastruktur-IAM-Rolle zu verwenden, müssen Sie Ihrem Benutzer die Erlaubnis erteilen, die Rolle an Amazon ECS weiterzugeben. Ordnen Sie Ihrem Benutzer die folgende `iam:PassRole` Berechtigung zu. Ersetzen Sie *`ecs InfrastructureRole`* durch den Namen der Infrastrukturrolle, die Sie erstellt haben.

```
{
    "Version": "2012-10-17",
    "Statement": [

        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInfrastructureRole"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Weitere Informationen zu Benutzerberechtigungen `iam:Passrole` und deren Aktualisierung finden Sie unter Gewähren von Benutzerberechtigungen zur Übergabe einer Rolle an einen AWS Dienst und Ändern der Berechtigungen für einen IAM-Benutzer im AWS Identity and Access Management Benutzerhandbuch.

Amazon ECS CodeDeploy IAM-Rolle

Bevor Sie den Bereitstellungstyp CodeDeploy Blau/Grün mit Amazon ECS verwenden können, benötigt der CodeDeploy Service Berechtigungen, um Ihren Amazon ECS-Service in Ihrem Namen zu aktualisieren. Diese Berechtigungen werden von der CodeDeploy IAM-Rolle () `ecsCodeDeployRole` bereitgestellt.

> **Note**
>
> Benutzer benötigen außerdem Nutzungsberechtigungen CodeDeploy. Diese Berechtigungen werden unter beschriebenErforderliche IAM-Berechtigungen.

Es werden zwei verwaltete Richtlinien bereitgestellt. Weitere Informationen finden Sie in einer der folgenden Informationen im Referenzhandbuch für AWS verwaltete Richtlinien:

- AWSCodeDeployRoleForECS- erteilt die CodeDeploy Erlaubnis, jede Ressource mithilfe der zugehörigen Aktion zu aktualisieren.
- AWSCodeDeployRoleForECSLimited- gibt CodeDeploy eingeschränktere Berechtigungen.

Die CodeDeploy Rolle erstellen

Sie können die folgenden Verfahren verwenden, um eine CodeDeploy Rolle für Amazon ECS zu erstellen.

AWS Management Console

Um die Servicerolle für CodeDeploy (IAM-Konsole) zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Rollen, und wählen Sie dann Rolle erstellen.
3. Wählen Sie für Vertrauenswürdige Entität die Option AWS-Service aus.
4. Wählen CodeDeploySie für Service oder Anwendungsfall die Option CodeDeploy - ECS-Anwendungsfall aus.
5. Wählen Sie Weiter aus.
6. Vergewissern Sie sich, dass im Abschnitt Richtlinie zum Anhängen von Berechtigungen die AWSCodeDeployRoleForECSRichtlinie ausgewählt ist.
7. Wählen Sie Weiter aus.
8. Geben Sie als Rollenname ecs CodeDeploy Role ein.
9. Prüfen Sie die Rolle und klicken Sie dann auf Create Role (Rolle erstellen).

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei mit dem Namen`codedeploy-trust-policy.json`, die die Vertrauensrichtlinie enthält, die für die CodeDeploy IAM-Rolle verwendet werden soll.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": ["codedeploy.amazonaws.com"]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Erstellen Sie eine IAM-Rolle namens `ecsCodedeployRole`, die im vorherigen Schritt erstellte Vertrauensrichtlinie verwendet.

```
aws iam create-role \
      --role-name ecsCodedeployRole \
      --assume-role-policy-document file://codedeploy-trust-policy.json
```

3. Hängen Sie die `AWSCodeDeployRoleForECS` oder die `AWSCodeDeployRoleForECSLimited` verwaltete Richtlinie an die `ecsTaskRole` Rolle an.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws::iam::aws:policy/AWSCodeDeployRoleForECS
```

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws::iam::aws:policy/AWSCodeDeployRoleForECSLimited
```

Wenn für die Aufgaben in Ihrem Service eine Aufgabenausführungsrolle erforderlich ist, müssen Sie der Rolle die `iam:PassRole` Berechtigung für jede Aufgabenausführungsrolle oder Aufgabenrollen-Außerkraftsetzung als CodeDeploy Richtlinie hinzufügen.

Berechtigungen für Rollen zur Aufgabenausführung

Wenn für die Aufgaben in Ihrem Service eine Aufgabenausführungsrolle erforderlich ist, müssen Sie der Rolle als `iam:PassRole` Richtlinie die Berechtigung für jede Aufgabenausführungsrolle oder Aufgabenrollenüberschreibung hinzufügen. CodeDeploy Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung und IAM-Rolle für Amazon ECS-Aufgaben. Anschließend fügen Sie diese Richtlinie der Rolle hinzu CodeDeploy

Erstellen der -Richtlinie

AWS Management Console

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Nachdem Sie die Richtlinie erstellt haben, fügen Sie die Richtlinie der CodeDeploy Rolle hinzu. Informationen zum Anhängen der Richtlinie an die Rolle finden Sie unter Ändern einer Rollenberechtigungsrichtlinie (Konsole) im AWS Identity and Access Management Benutzerhandbuch.

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei mit dem Namen `blue-green-iam-passrole.json` und folgendem Inhalt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

2. Verwenden Sie den folgenden Befehl, um die IAM-Richtlinie mithilfe der JSON-Richtliniendokumentdatei zu erstellen.

```
aws iam create-policy \
      --policy-name cdTaskExecutionPolicy \
      --policy-document file://blue-green-iam-passrole.json
```

3. Rufen Sie den ARN der IAM-Richtlinie ab, die Sie mit dem folgenden Befehl erstellt haben.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cdTaskExecutionPolicy`].Arn'
```

4. Verwenden Sie den folgenden Befehl, um die Richtlinie an die CodeDeploy IAM-Rolle anzuhängen.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cdTaskExecutionPolicy
```

Amazon ECS EventBridge IAM-Rolle

Bevor Sie geplante Amazon ECS-Aufgaben mit EventBridge Regeln und Zielen verwenden können, benötigt der EventBridge Service Berechtigungen, um Amazon ECS-Aufgaben in Ihrem Namen auszuführen. Diese Berechtigungen werden von der EventBridge IAM-Rolle (`ecsEventsRole`) bereitgestellt.

Die Richtlinie `AmazonEC2ContainerServiceEventsRole` wird unten gezeigt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
```

```
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

Wenn Ihre geplanten Aufgaben die Verwendung einer Aufgabenausführungsrolle, einer Aufgabenrolle oder einer Aufgabenrollenüberschreibung erfordern, müssen Sie der EventBridge IAM-Rolle `iam:PassRole` Berechtigungen für jede Aufgabenausführungsrolle, Aufgabenrolle oder Aufgabenrollenüberschreibung hinzufügen. Weitere Informationen zur Aufgabenausführungsrolle finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.

> **Note**
>
> Geben Sie den vollständigen ARN Ihrer Aufgabenausführungsrolle oder Ihres Aufgabenrollen-Overrides an.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

Sie können wählen, ob Sie die EventBridge Rolle für Sie AWS Management Console erstellen lassen möchten, wenn Sie eine geplante Aufgabe konfigurieren. Weitere Informationen finden Sie unter Verwenden von Amazon EventBridge Scheduler zur Planung von Amazon ECS-Aufgaben .

Die EventBridge Rolle wird erstellt

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei namens `eventbridge-trust-policy.json`, die die Vertrauensrichtlinie enthält, die für die IAM-Rolle verwendet werden soll. Die Datei sollte Folgendes enthalten:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Verwenden Sie den folgenden Befehl, um eine IAM-Rolle zu erstellen, die `ecsEventsRole` mithilfe der Vertrauensrichtlinie benannt wird, die Sie im vorherigen Schritt erstellt haben.

```
aws iam create-role \
      --role-name ecsEventsRole \
      --assume-role-policy-document file://eventbridge-policy.json
```

3. Hängen Sie die AWS verwaltete `AmazonEC2ContainerServiceEventsRole` Datei mit dem folgenden Befehl an die `ecsEventsRole` Rolle an.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceEventsRole
```

Sie können die Rolle auch mithilfe des Workflows für benutzerdefinierte Vertrauensrichtlinien (https://console.aws.amazon.com/iam/) der IAM-Konsole erstellen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer Rolle mithilfe benutzerdefinierter Vertrauensrichtlinien (Konsole).

Anfügen einer Richtlinie an eine **`ecsEventsRole`**-Rolle

Sie können die folgenden Verfahren verwenden, um der EventBridge IAM-Rolle Berechtigungen für die Aufgabenausführungsrolle hinzuzufügen.

AWS Management Console

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Nachdem Sie die Richtlinie erstellt haben, fügen Sie die Richtlinie der EventBridge Rolle hinzu. Informationen zum Anhängen der Richtlinie an die Rolle finden Sie unter Ändern einer Rollenberechtigungsrichtlinie (Konsole) im AWS Identity and Access Management Benutzerhandbuch.

AWS CLI

Ersetzen Sie alle *Benutzereingaben* durch Ihre eigenen Informationen.

1. Erstellen Sie eine Datei mit dem Namen `ev-iam-passrole.json` und folgendem Inhalt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

2. Verwenden Sie den folgenden AWS CLI Befehl, um die IAM-Richtlinie mithilfe der JSON-Richtliniendokumentdatei zu erstellen.

```
aws iam create-policy \
      --policy-name eventsTaskExecutionPolicy \
      --policy-document file://ev-iam-passrole.json
```

3. Rufen Sie den ARN der IAM-Richtlinie ab, die Sie mit dem folgenden Befehl erstellt haben.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`eventsTaskExecutionPolicy`].Arn'
```

4. Verwenden Sie den folgenden Befehl, um die Richtlinie mithilfe des EventBridge Richtlinien-ARN an die IAM-Rolle anzuhängen.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/eventsTaskExecutionPolicy
```

Erforderliche Berechtigungen für die Amazon-ECS-Konsole

Nach den bewährten Methoden zur Erteilung von geringsten Privilegien können Sie die von `AmazonECS_FullAccess` verwaltete Richtlinie als Vorlage zum Erstellen eigener benutzerdefinierter Richtlinien verwenden. Auf diese Weise können Sie Berechtigungen zu und aus der verwalteten Richtlinie basierend auf Ihren spezifischen Anforderungen entfernen oder hinzufügen. Weitere Informationen finden Sie unter Details zu Berechtigungen.

Die Amazon ECS-Konsole wird betrieben von AWS CloudFormation und benötigt in den folgenden Fällen zusätzliche IAM-Berechtigungen:

- Erstellen eines Clusters
- Erstellen eines Service
- Erstellen eines Kapazitätsanbieters

Sie können eine Richtlinie für die zusätzlichen Berechtigungen erstellen und sie dann der IAM-Rolle zuordnen, die Sie für den Zugriff auf die Konsole verwenden. Weitere Informationen finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Für die Erstellung eines Clusters erforderliche Berechtigungen

Wenn Sie einen Cluster in der Konsole erstellen, benötigen Sie zusätzliche Berechtigungen, die Ihnen Berechtigungen zur Verwaltung von AWS CloudFormation Stacks gewähren.

Die folgende zusätzliche Berechtigung ist erforderlich:

- `cloudformation`: Ermöglicht Prinzipalen das Erstellen und Verwalten von AWS CloudFormation -Stapeln. Dies ist erforderlich, wenn Sie Amazon-ECS-Cluster mit AWS Management Console erstellen und anschließend dieser Cluster verwalten.

Die folgende Richtlinie enthält die erforderlichen AWS CloudFormation Berechtigungen und beschränkt die Aktionen auf Ressourcen, die in der Amazon ECS-Konsole erstellt wurden.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-Cluster-*"
            ]
      }
   ]
}
```

Wenn Sie die Container-Instance-Rolle (`ecsInstanceRole`) von Amazon ECS nicht erstellt haben und einen Cluster erstellen, der Amazon-EC2-Instances verwendet, erstellt die Konsole die Rolle in Ihrem Namen.

Wenn Sie Auto Scaling Scaling-Gruppen verwenden, benötigen Sie außerdem zusätzliche Berechtigungen, damit die Konsole den Auto Scaling-Gruppen Tags hinzufügen kann, wenn Sie die Cluster-Auto-Scaling-Funktion verwenden.

Die folgende zusätzliche Berechtigung ist erforderlich:

- `autoscaling` – Ermöglicht es der Konsole, eine Amazon-EC2-Auto-Scaling-Gruppe mit einem Tag zu versehen. Dies ist erforderlich, wenn Amazon-EC2-Auto-Scaling-Gruppen bei der Verwendung des Features für das Auto Scaling von Clustern verwaltet werden. Das Tag ist das von ECS verwaltete Tag, das die Konsole der Gruppe automatisch hinzufügt, um anzuzeigen, dass es in der Konsole erstellt wurde.
- `iam`: Ermöglicht es Prinzipalen, IAM-Rollen und ihre angehängten Richtlinien aufzulisten. Prinzipale können auch Instance-Profile auflisten, die für Ihre Amazon-EC2-Instances verfügbar sind.

Die folgende Richtlinie enthält die erforderlichen IAM-Berechtigungen und beschränkt die Aktionen auf die `ecsInstanceRole`-Rolle.

Die Auto-Scaling-Berechtigungen sind nicht begrenzt.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:CreateOrUpdateTags",
            "Resource": "*"
        }
    ]
}
```

Für die Erstellung eines Kapazitätsanbieters sind Berechtigungen erforderlich

Wenn Sie einen Dienst in der Konsole erstellen, benötigen Sie zusätzliche Berechtigungen, die Ihnen Berechtigungen zur Verwaltung von AWS CloudFormation Stacks gewähren. Die folgende zusätzliche Berechtigung ist erforderlich:

- `cloudformation`: Ermöglicht Prinzipalen das Erstellen und Verwalten von AWS CloudFormation -Stapeln. Dies ist für die Erstellung von Amazon-ECS-Kapazitätsanbietern mit der AWS Management Console und die anschließende Verwaltung dieser Kapazitätsanbieter erforderlich.

Die folgende Richtlinie enthält die erforderlichen Berechtigungen und beschränkt die Aktionen auf Ressourcen, die in der Amazon-ECS-Konsole erstellt wurden.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-CapacityProvider-*"
            ]
      }
   ]
}
```

Für die Erstellung eines Dienstes sind Berechtigungen erforderlich

Wenn Sie einen Dienst in der Konsole erstellen, benötigen Sie zusätzliche Berechtigungen, die Ihnen Berechtigungen zur Verwaltung von AWS CloudFormation Stacks gewähren. Die folgende zusätzliche Berechtigung ist erforderlich:

- `cloudformation`: Ermöglicht Prinzipalen das Erstellen und Verwalten von AWS CloudFormation -Stapeln. Dies ist für die Erstellung und anschließende Verwaltung von Amazon-ECS-Services mit der AWS Management Console erforderlich.

Die folgende Richtlinie enthält die erforderlichen Berechtigungen und beschränkt die Aktionen auf Ressourcen, die in der Amazon-ECS-Konsole erstellt wurden.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
            ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/ECS-Console-V2-Service-*"
            ]
      }
   ]
}
```

Berechtigungen für die Erstellung von IAM-Rollen

Für die folgenden Aktionen sind zusätzliche Berechtigungen erforderlich, um den Vorgang abzuschließen:

- Registrieren einer externen Instance – weitere Informationen finden Sie unter IAM-Rolle in Amazon ECS Anywhere
- Registrieren einer Aufgabendefinition – für weitere Informationen erhalten Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung
- Eine EventBridge Regel zur Planung von Aufgaben erstellen — weitere Informationen finden Sie unter Amazon ECS EventBridge IAM-Rolle

Sie können diese Berechtigungen hinzufügen, indem Sie eine Rolle in IAM erstellen, bevor Sie sie in der Amazon-ECS-Konsole verwenden. Wenn Sie die Rollen nicht erstellen, erstellt die Amazon-ECS-Konsole sie in Ihrem Namen.

Erforderliche Berechtigungen für die Registrierung einer externen Instanz in einem Cluster

Sie benötigen zusätzliche Berechtigungen, wenn Sie eine externe Instance in einem Cluster registrieren und eine neue externe Instance-Rolle (`escExternalInstanceRole`) erstellen möchten.

Die folgende zusätzliche Berechtigung ist erforderlich:

- `iam` – Ermöglicht es Prinzipalen, IAM-Rollen und ihre angehängten Richtlinien zu erstellen und aufzulisten.
- `ssm` – Ermöglicht es Prinzipalen, die externe Instance bei Systems Manager zu registrieren.

> **Note**
>
> Um eine vorhandene `escExternalInstanceRole` auswählen zu können, benötigen Sie die Berechtigungen `iam:GetRole` und `iam:PassRole`.

Die folgende Richtlinie enthält die erforderlichen Berechtigungen und beschränkt die Aktionen auf die `escExternalInstanceRole`-Rolle.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/escExternalInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": ["iam:PassRole","ssm:CreateActivation"],
            "Resource": "arn:aws:iam::*:role/escExternalInstanceRole"
        }
    ]
```

```
}
```

Für die Registrierung einer Aufgabendefinition sind Berechtigungen erforderlich

Sie benötigen zusätzliche Berechtigungen, wenn Sie eine Aufgabendefinition registrieren und eine neue Aufgabenausführungsrolle (`ecsTaskExecutionRole`) erstellen möchten.

Die folgende zusätzliche Berechtigung ist erforderlich:

- `iam` – Ermöglicht es Prinzipalen, IAM-Rollen und ihre angehängten Richtlinien zu erstellen und aufzulisten.

> **Note**
>
> Um eine vorhandene `ecsTaskExecutionRole` auswählen zu können, benötigen Sie die Berechtigung `iam:GetRole`.

Die folgende Richtlinie enthält die erforderlichen Berechtigungen und beschränkt die Aktionen auf die `ecsTaskExecutionRole`-Rolle.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsTaskExecutionRole"
        }
    ]
}
```

Für die Erstellung einer EventBridge Regel für geplante Aufgaben sind Berechtigungen erforderlich

Sie benötigen zusätzliche Berechtigungen, wenn Sie eine Aufgabe planen und eine neue Rolle der Rolle „ CloudWatch Ereignisse“ (`ecsEventsRole`) erstellen möchten.

Die folgende zusätzliche Berechtigung ist erforderlich:

- `iam` – Ermöglicht es Prinzipalen, IAM-Rollen und die zugehörigen Richtlinien zu erstellen und aufzulisten und es Amazon ECS zu ermöglichen, die Rolle an andere Services weiterzugeben, damit diese Rolle übernommen wird.

> **Note**
>
> Um eine vorhandene `ecsEventsRole` auswählen zu können, benötigen Sie die Berechtigungen `iam:GetRole` und `iam:PassRole`.

Die folgende Richtlinie enthält die erforderlichen Berechtigungen und beschränkt die Aktionen auf die `ecsEventsRole`-Rolle.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole",
              "iam: PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsEventsRole"
        }
    ]
}
```

Für Amazon ECS Service Auto Scaling sind IAM-Berechtigungen erforderlich

Service Auto Scaling wird durch eine Kombination der Amazon ECS- und Application Auto Scaling Scaling-APIs ermöglicht. CloudWatch Services werden mit Amazon ECS erstellt und aktualisiert, Alarme werden mit CloudWatch Application Auto Scaling erstellt und Skalierungsrichtlinien werden erstellt.

Zusätzlich zu den standardmäßigen IAM-Berechtigungen für das Erstellen und Aktualisieren von Diensten sind die folgenden Berechtigungen erforderlich, um mit den Service Auto Scaling Scaling-Einstellungen zu interagieren, wie in der folgenden Beispielrichtlinie dargestellt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ecs:DescribeServices",
                "ecs:UpdateService",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Die Beispiel für einen Amazon ECS-Service erstellen- und Beispiel für einen Amazon ECS-Service aktualisieren-IAM-Richtlinienbeispiele zeigen die erforderlichen Berechtigungen zur Verwendung von Service Auto Scaling in der AWS Management Console.

Der Application Auto Scaling-Service benötigt außerdem die Erlaubnis, Ihre Amazon ECS-Services und CloudWatch -Alarme zu beschreiben, sowie die Erlaubnis, die von Ihrem Service gewünschte Anzahl in Ihrem Namen zu ändern. Die `sns:` Berechtigungen gelten für Benachrichtigungen, die an ein Amazon SNS SNS-Thema CloudWatch gesendet werden, wenn ein Schwellenwert überschritten wurde. Wenn Sie das Auto Scaling für Ihre Amazon-ECS-Services benutzen, wird eine serviceverknüpfte Rolle namens `AWSServiceRoleForApplicationAutoScaling_ECSService` erstellt. Diese serviceverknüpfte Rolle gewährt die Application Auto Scaling-Berechtigung, die Alarme für Ihre Richtlinien zu beschreiben, die aktuelle Anzahl der laufenden Aufgaben des Services zu

überwachen und die gewünschte Anzahl des Services zu ändern. Die ursprüngliche verwaltete Amazon-ECS-Rolle für Application Auto Scaling war `ecsAutoscaleRole`, diese wird jedoch nicht mehr benötigt. Die Service-verknüpfte Rolle ist die Standardrolle für Application Auto Scaling. Weitere Informationen finden Sie unter Serviceverknüpfte Rollen für Application Auto Scaling im Benutzerhandbuch zu Application Auto Scaling.

Wenn Sie Ihre Amazon ECS-Container-Instance-Rolle erstellt haben, bevor CloudWatch Metriken für Amazon ECS verfügbar waren, müssen Sie möglicherweise die `ecs:StartTelemetrySession` Berechtigung hinzufügen. Weitere Informationen finden Sie unter Überlegungen.

Berechtigung zum Markieren von Ressourcen bei der Erstellung gewähren

Mit den folgenden Amazon-ECS-API-Aktionen zum Erstellen einer Amazon ECS-API können Sie Tags beim Erstellen der Ressource angeben. Wenn bei der Aktion zur Erstellung von Ressourcen Tags angegeben wurden, AWS führt eine zusätzliche Autorisierung durch, um zu überprüfen, ob die richtigen Berechtigungen zum Erstellen von Tags zugewiesen wurden.

- `CreateCapacityProvider`
- `CreateCluster`
- `CreateService`
- `CreateTaskSet`
- `RegisterContainerInstance`
- `RegisterTaskDefinition`
- `RunTask`
- `StartTask`

Sie können Resource-Tags (Markierungen) verwenden, um eine attributbasierte Steuerung (ABAC) zu implementieren. Weitere Informationen finden Sie unter the section called “Steuerung des Zugriffs auf Amazon-ECS-Ressourcen mithilfe von Ressourcen-Tags” und Markieren von Ressourcen.

Um Tags bei der Erstellung zuzulassen, erstellen oder ändern Sie eine Richtlinie, die sowohl die Berechtigungen zur Verwendung der Aktion, mit der die Ressource erstellt wird, wie z. B. `ecs:CreateCluster` oder `ecs:RunTask` und die Aktion `ecs:TagResource` enthält.

Das folgende Beispiel zeigt eine Richtlinie, die es Benutzern ermöglicht, während der Clustererstellung Cluster zu erstellen und Tags hinzuzufügen. Die Markierung von bestehenden

Ressourcen durch die Benutzer ist nicht zulässig. (Sie können die `ecs:TagResource`-Aktion nicht direkt aufrufen.)

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:CreateCluster"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
                  "ecs:CreateAction": [
                      "CreateCluster",
                      "CreateCapacityProvider",
                      "CreateService",
                      "CreateTaskSet",
                      "RegisterContainerInstance",
                      "RegisterTaskDefinition",
                      "RunTask",
                      "StartTask"
                  ]
          }
       }
    }
  ]
}
```

Die `ecs:TagResource`-Aktion wird nur ausgewertet, wenn die Tags während der Aktion zur Ressourcenerstellung angewendet werden. Folglich benötigt ein Benutzer, der über die Berechtigungen zum Erstellen einer Ressource verfügt (vorausgesetzt, es bestehen keine Markierungsbedingungen), keine Berechtigungen zur Verwendung der `ecs:TagResource`-Aktion, wenn keine Tags in der Anforderung angegeben werden. Wenn der Benutzer allerdings versucht,

eine Ressource mit Tags zu erstellen, schlägt die Anforderung fehl, wenn der Benutzer nicht über die Berechtigungen für die `ecs:TagResource`-Aktion verfügt.

Amazon ECS steuert den Zugriff auf bestimmte Tags

Sie können zusätzliche Bedingungen im `Condition`-Element Ihrer IAM-Richtlinien verwenden, um die Tag-Schlüssel und -Werte zu steuern, die auf Ressourcen angewendet werden können.

Die folgenden Bedingungsschlüssel können mit den Beispielen im vorangegangenen Abschnitt verwendet werden:

- `aws:RequestTag`: Gibt an, dass eine Anforderung einen bestimmten Tag-Schlüssel oder Tag-Schlüssel und -Wert enthalten muss. In der Anforderung können auch andere Tags (Markierungen) angegeben werden.
  - Zusammen mit dem `StringEquals`-Bedingungsoperator können Sie eine bestimmte Tag-Schlüssel- und -Wert-Kombination erzwingen, z. B. den Tag `cost-center=cc123`:

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - In Kombination mit dem `StringLike`-Bedingungsoperator erzwingen Sie einen bestimmten Tag-Schlüssel in der Anforderung, beispielsweise den Tag-Schlüssel `purpose`:

    ```
    "StringLike": { "aws:RequestTag/purpose": "*" }
    ```

- `aws:TagKeys`: Erzwingt die Tag-Schlüssel, die in der Anforderung verwendet werden.
  - Verwenden Sie diesen Bedingungsschlüssel mit dem `ForAllValues`-Modifikator, um bestimmte Tag-Schlüssel zu erzwingen, die in der Anforderung bereitgestellt werden (wenn in der Anforderung Tags angegeben werden, sind nur bestimmte Tags erlaubt und keine anderen Tags gestattet). Im folgenden Beispiel sind die Tag-Schlüssel `environment` oder `cost-center` erlaubt:

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - Zusammen mit dem `ForAnyValue`-Modifikator wird erzwungen, dass die Anforderung mindestens einen der angegebenen Tag-Schlüssel umfassen muss. Zum Beispiel muss mindestens einer der Tag-Schlüssel `environment` und `webserver` in der Anforderung enthalten sein:

    ```
    "ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
    ```

Diese Bedingungsschlüssel können auf Aktionen zur Ressourcenerstellung, die Tagging unterstützen, sowie auf die Aktion `ecs:TagResource` angewendet werden. Informationen darüber, ob eine Amazon-ECS-API-Aktion das Tagging unterstützt, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon ECS.

Wenn Sie erzwingen möchten, dass die Benutzer beim Erstellen einer Ressource Tags angeben, müssen Sie den `aws:RequestTag`-Bedingungsschlüssel oder den `aws:TagKeys`-Bedingungsschlüssel mit dem `ForAnyValue`-Modifikator für die Aktion zur Erstellung von Ressourcen verwenden. Wenn ein Benutzer für diese Aktion zur Ressourcenerstellung keine Tags angibt, wird die `ecs:TagResource`-Aktion nicht ausgewertet.

Bei Bedingungen gilt, dass die Groß- und Kleinschreibung für den Bedingungsschlüssel nicht berücksichtigt und für den Bedingungswert beachtet wird. Verwenden Sie aus diesem Grund den `aws:TagKeys`-Bedingungsschlüssel und geben Sie den Tag (Markierung)-Schlüssel als Wert dieser Bedingung an, wenn Sie die Berücksichtigung der Groß- und Kleinschreibung für einen Tag (Markierung)-Schlüssel erzwingen möchten.

Weitere Informationen zu Bedingungen mit mehreren Werten erhalten Sie unter Erstellen einer Bedingung zum Testen von mehreren Schlüsselwerten im IAM-Benutzerhandbuch.

Steuerung des Zugriffs auf Amazon-ECS-Ressourcen mithilfe von Ressourcen-Tags

Wenn Sie eine IAM-Richtlinie erstellen, die Benutzern die Berechtigung zur Verwendung von Amazon-ECS-Ressourcen gewährt, können Sie Tag-Informationen in das `Condition`-Element der Richtlinie einfügen, um den Zugriff basierend auf Tags zu steuern. Dies wird als attributbasierte Zugriffskontrolle (ABAC) bezeichnet. ABAC bietet eine besser Kontrolle darüber, welche Ressourcen ein Benutzer ändern, verwenden oder löschen kann. Weitere Informationen finden Sie unter Was ist ABAC für AWS?

Sie können z. B. eine Richtlinie erstellen, die es Benutzern erlaubt, einen Cluster zu löschen, diese Aktion aber verweigert, wenn der Cluster das Tag `environment=production` hat. Dazu verwenden Sie den `aws:ResourceTag`-Bedingungsschlüssel, um den Zugriff auf die Ressource basierend auf den der Ressource zugewiesenen Tags (Markierung) zu erlauben oder zu verweigern.

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Informationen darüber, ob eine Amazon ECS-API-Aktion das Steuern des Zugriffs mithilfe des `aws:ResourceTag`-Bedingungsschlüssels unterstützt finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon ECS. Beachten Sie, dass die `Describe`-Aktionen keine

Berechtigungen auf Ressourcenebene unterstützen, sodass sie in einer separaten Anweisung ohne Bedingungen angegeben werden müssen.

Beispiele für IAM-Richtlinien finden Sie unter Amazon ECS-Beispielrichtlinien .

Wenn Sie Benutzern den Zugriff zu Ressourcen auf der Grundlage von Tags (Markierungen) gewähren oder verweigern, müssen Sie daran denken, Benutzern explizit das Hinzufügen und Entfernen dieser Tags (Markierungen) von den jeweiligen Ressourcen unmöglich zu machen. Andernfalls können Benutzer möglicherweise Ihre Einschränkungen umgehen und sich Zugriff auf eine Ressource verschaffen, indem sie ihre Tags (Markierungen) modifizieren.

Amazon ECS-Beispielrichtlinien

Mit IAM-Richtlinien können Sie Benutzern Berechtigungen erteilen, um bestimmte Ressourcen in der Amazon-ECS-Konsole anzusehen und mit diesen zu arbeiten. Sie können die Beispielrichtlinien aus dem vorherigen Abschnitt verwenden. Sie sind jedoch für Anfragen konzipiert, die mit dem AWS CLI oder einem AWS SDK gestellt werden.

Beispiel: Benutzern das Löschen eines Amazon ECS-Clusters auf der Grundlage von Tags ermöglichen

Die folgende Richtlinie ermöglicht es Benutzern, Cluster zu löschen, wenn das Tag das Schlüssel/Wert-Paar „Purpose/Testing“ enthält.

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ecs:region:account-id:cluster/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Testing"
        }
      }
    }
  ]
}
```

Identitäts- und Zugriffsfehler bei Amazon Elastic Container Service

Diagnostizieren und beheben Sie mithilfe der folgenden Informationen gängige Probleme, die bei der Verwendung von Amazon ECS und IAM auftreten können.

Themen

Ich bin nicht autorisiert, eine Aktion in Amazon ECS auszuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `ecs:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ecs:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `ecs:`*`GetWidget`*-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam durchzuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zur Ausführung der Aktion „`iam:PassRole`“ autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an Amazon ECS übergeben zu können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon ECS auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine Amazon ECS-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen dazu, ob Amazon ECS diese Features unterstützt, finden Sie unter So funktioniert Amazon Elastic Container Service mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, über AWS-Konten die Sie verfügen, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

Weitere Ressourcen zur Fehlerbehebung

Auf den folgenden Seiten finden Sie Informationen zu Fehlercodes:

- Fehlermeldungen zum Abbruch von Aufgaben durch Amazon ECS
- Ereignismeldungen des Amazon ECS-Service anzeigen

Bewährte IAM-Methoden für Amazon ECS

Sie können AWS Identity and Access Management (IAM) verwenden, um den Zugriff auf Ihre AWS Dienste und Ressourcen mithilfe von regelbasierten Richtlinien für Authentifizierungs- und Autorisierungszwecke zu verwalten und zu kontrollieren. Insbesondere kontrollieren Sie mit diesem Service den Zugriff auf Ihre AWS Ressourcen mithilfe von Richtlinien, die auf Benutzer, Gruppen oder Rollen angewendet werden. Unter diesen drei sind Benutzer die Konten, die Zugriff auf Ihre Ressourcen haben können. Und eine IAM-Rolle besteht aus einer Reihe von Berechtigungen, die von einer authentifizierten Identität übernommen werden können, die keiner bestimmten Identität außerhalb von IAM zugeordnet ist. Weitere Informationen finden Sie unter Amazon ECS im Überblick über die Zugriffsverwaltung: Berechtigungen und Richtlinien.

Sich an die Richtlinie mit dem Zugriff mit der geringsten Berechtigung halten

Erstellen Sie Richtlinien, die so begrenzt sind, dass Benutzer ihre vorgeschriebenen Aufgaben ausführen können. Wenn ein Entwickler beispielsweise eine Aufgabe regelmäßig beenden muss, erstellen Sie eine Richtlinie, die nur diese bestimmte Aktion zulässt. Das folgende Beispiel erlaubt es einem Benutzer nur, eine Aufgabe zu beenden, die zu einer bestimmten `task_family` auf einem Cluster mit einem bestimmten Amazon-Ressourcenname (ARN) gehört. Der Verweis auf einen ARN in einer Bedingung ist auch ein Beispiel für die Verwendung von Berechtigungen auf Ressourcenebene. Sie können Berechtigungen auf Ressourcenebene verwenden, um die Ressource anzugeben, auf die eine Aktion angewendet werden soll.

> **Note**
>
> Wenn Sie in einer Richtlinie auf einen ARN verweisen, verwenden Sie das neue längere ARN-Format. Weitere Informationen finden Sie unter Amazon-Ressourcenname (ARN) und IDs im Entwicklerhandbuch für Amazon Elastic Container Service.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:StopTask"
      ],
      "Condition": {
        "ArnEquals": {
          "ecs:cluster": "arn:aws:ecs:region:account_id:cluster/cluster_name"
        }
      },
      "Resource": [
        "arn:aws:ecs:region:account_id:task-definition/task_family:*"
      ]
    }
  ]
}
```

Lassen Sie Cluster-Ressourcen als administrative Grenze dienen

Richtlinien, die zu eng gefasst sind, können zu einer Vielzahl von Rollen führen und den Verwaltungsaufwand erhöhen. Anstatt Rollen zu erstellen, die nur auf bestimmte Aufgaben oder Services beschränkt sind, sollten Sie Rollen erstellen, die auf Cluster zugeschnitten sind, und den Cluster als primäre administrative Grenze verwenden.

Erstellen Sie automatisierte Pipelines, um Endbenutzer von der API zu isolieren

Sie können die Aktionen einschränken, die Benutzer verwenden können, indem Sie Pipelines erstellen, die Anwendungen automatisch verpacken und auf Amazon-ECS-Clustern bereitstellen. Dadurch wird das Erstellen, Aktualisieren und Löschen von Aufgaben effektiv an die Pipeline delegiert. Weitere Informationen finden Sie unter Tutorial: Amazon ECS-Standardbereitstellung mit CodePipeline im AWS CodePipeline Benutzerhandbuch.

Richtlinienbedingungen für zusätzliche Sicherheit verwenden

Wenn Sie eine zusätzliche Sicherheitsebene benötigen, fügen Sie Ihrer Richtlinie eine Bedingung hinzu. Dies kann nützlich sein, wenn Sie einen privilegierten Vorgang ausführen oder wenn Sie die Anzahl der Aktionen einschränken müssen, die für bestimmte Ressourcen ausgeführt werden können. Die folgende Beispielrichtlinie erfordert eine mehrstufige Autorisierung beim Löschen eines Clusters.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

Auf Services angewendete Tags werden auf alle Aufgaben übertragen, die Teil dieses Services sind. Aus diesem Grund können Sie Rollen erstellen, die auf Amazon-ECS-Ressourcen mit bestimmten Tags beschränkt sind. In der folgenden Richtlinie startet und stoppt ein IAM-Prinzipal alle Aufgaben mit einem Tag-Schlüssel von `Department` und einem Tag-Wert von `Accounting`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Überprüfen Sie regelmäßig den Zugriff auf die APIs

Ein Benutzer könnte die Rollen wechseln. Nach einem Rollenwechsel gelten die Berechtigungen, die ihnen zuvor gewährt wurden, möglicherweise nicht mehr. Stellen Sie sicher, dass Sie prüfen, wer Zugriff auf die Amazon-ECS-APIs hat und ob dieser Zugriff weiterhin gewährleistet ist. Erwägen Sie die Integration von IAM mit einer Lösung für das Benutzerlebenszyklusmanagement, die den Zugriff automatisch widerruft, wenn ein Benutzer das Unternehmen verlässt. Weitere Informationen finden Sie in den Amazon-ECS-Sicherheitsaudit-Richtlinien im Allgemeine Amazon Web Services-Referenz.

Protokollierung und Überwachung in Amazon Elastic Container Service

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Elastic Container Service und Ihrer AWS Lösungen. Sie sollten Überwachungsdaten aus allen Teilen Ihrer AWS Lösung sammeln, damit Sie einen etwaigen Fehler an mehreren Stellen leichter debuggen können. AWS bietet verschiedene Tools zur Überwachung Ihrer Amazon ECS-Ressourcen und zur Reaktion auf potenzielle Vorfälle:

CloudWatch Amazon-Alarme

Überwachen Sie eine Metrik über einen bestimmten, von Ihnen definierten Zeitraum und führen Sie einzelne oder mehrere Aktionen durch, die vom Wert der Metrik im Vergleich zu einem festgelegten Schwellenwert in einer Reihe von Zeiträumen abhängen. Die Aktion ist eine Benachrichtigung, die an ein Amazon Simple Notification Service (Amazon SNS) -Thema oder eine Amazon EC2 Auto Scaling Scaling-Richtlinie gesendet wird. CloudWatch Alarme lösen keine Aktionen aus, nur weil sie sich in einem bestimmten Status befinden. Der Status muss sich geändert haben und für eine bestimmte Anzahl von Zeiträumen beibehalten worden sein. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS mit CloudWatch .

Für Dienste mit Aufgaben, die den Starttyp Fargate verwenden, können Sie CloudWatch Alarme verwenden, um die Aufgaben in Ihrem Service auf der Grundlage von CloudWatch Metriken wie CPU- und Speicherauslastung zu vergrößern und zu skalieren. Weitere Informationen finden Sie unter Skalieren Sie Ihren Amazon ECS-Service automatisch.

Bei Clustern mit Aufgaben oder Diensten, die den EC2-Starttyp verwenden, können Sie CloudWatch Alarme verwenden, um die Container-Instances auf der Grundlage von CloudWatch Metriken wie der Cluster-Speicherreservierung zu vergrößern und zu skalieren.

CloudWatch Amazon-Protokolle

Sie können die Protokolldateien über die Container in Ihren Amazon-ECS-Aufgaben überwachen, speichern und darauf zugreifen, indem Sie den Protokolltreiber `awslogs` in Ihren Aufgabendefinitionen angeben. Weitere Informationen finden Sie unter Verwenden von awslogs driver.

Sie können von Ihren Amazon-ECS-Container-Instances aus die Betriebssystem- und Amazon-ECS-Container-Agent-Protokolldateien überwachen, speichern und darauf zugreifen. Diese Methode für den Zugriff auf Protokolle kann für Container verwendet werden, die den EC2-Starttyp verwenden.

CloudWatch Amazon-Veranstaltungen

Ordnen Sie Ereignisse zu und leiten Sie sie zu einer oder mehreren Zielfunktionen oder Streams um, um Änderungen vorzunehmen, Statusinformationen zu erfassen und Korrekturmaßnahmen durchzuführen. Weitere Informationen finden Sie Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge in diesem Leitfaden und unter Was ist Amazon CloudWatch Events? im Amazon CloudWatch Events-Benutzerhandbuch.

AWS CloudTrail Logs

CloudTrail bietet eine Aufzeichnung der Aktionen, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon ECS ausgeführt wurden. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon ECS gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln. Weitere Informationen finden Sie unter Amazon ECS-API-Aufrufe protokollieren mit AWS CloudTrail.

AWS Trusted Advisor

Trusted Advisor stützt sich auf bewährte Verfahren, die wir bei der Betreuung von Hunderttausenden von AWS Kunden gelernt haben. Trusted Advisor untersucht Ihre AWS Umgebung und gibt dann Empfehlungen, wenn Möglichkeiten bestehen, Geld zu sparen, die Systemverfügbarkeit und -leistung zu verbessern oder Sicherheitslücken zu schließen. Alle AWS Kunden haben Zugriff auf fünf Trusted Advisor Schecks. Kunden mit einem Business- oder Enterprise-Supportplan können alle Trusted Advisor Schecks einsehen.

Weitere Informationen finden Sie unter AWS Trusted Advisor im AWS Support - Benutzerhandbuch.

AWS Compute Optimizer

AWS Compute Optimizer ist ein Service, der die Konfiguration und Nutzungskennzahlen Ihrer AWS Ressourcen analysiert. Es berichtet, ob Ihre Ressourcen optimal sind und generiert Optimierungsempfehlungen, um die Kosten zu senken und die Leistung Ihrer Workloads zu verbessern.

Weitere Informationen finden Sie unter AWS Compute Optimizer Empfehlungen für Amazon ECS.

Ein weiterer wichtiger Teil der Überwachung von Amazon ECS ist die manuelle Überwachung der Elemente, die von den CloudWatch Alarmen nicht abgedeckt werden. Die Konsolen-Dashboards CloudWatch Trusted Advisor, und andere AWS Konsolen-Dashboards bieten einen at-a-glance Überblick über den Zustand Ihrer AWS Umgebung. Wir empfehlen, dass Sie auch die Protokolldateien auf Ihren Container-Instances und die Container in Ihren Aufgaben überprüfen.

Validierung der Compliance für Amazon Elastic Container Service

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- Schnellstartanleitungen zu Sicherheit und Compliance — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von Basisumgebungen beschrieben AWS , bei denen Sicherheit und Compliance im Mittelpunkt stehen.
- Architecting for HIPAA Security and Compliance on Amazon Web Services — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-fähige Anwendungen erstellen AWS können.

> **Note**
>
> AWS-Services Nicht alle sind HIPAA-fähig. Weitere Informationen finden Sie in der Referenz für HIPAA-berechtigte Services.

- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- AWS Leitfäden zur Einhaltung von Vorschriften für Kunden — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- Evaluierung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub— Auf diese AWS-Service Weise erhalten Sie einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerungsreferenz.
- Amazon GuardDuty — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- AWS Audit Manager— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

Bewährte Methoden zur Einhaltung von Vorschriften und Sicherheit für Amazon ECS

Ihre Compliance-Verantwortung bei Verwendung von Amazon ECS hängt von der Vertraulichkeit der Daten, den Compliance-Zielen des Unternehmens und den geltenden Gesetzen und Vorschriften ab.

AWS bietet die folgenden Ressourcen zur Unterstützung bei der Einhaltung von Vorschriften:

- Kurzanleitungen zu Sicherheit und Compliance: In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Implementierung von sicherheits- und Compliance-orientierten Basisumgebungen beschrieben. AWS
- Whitepaper „Architecting for HIPAA Security and Compliance“: In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-konforme Anwendungen entwickeln können. AWS
- AWS -Services im Umfang des Compliance Programms: Diese Liste enthält die AWS -Services, die zum Umfang bestimmter Compliance-Programme gehören. Weitere Informationen finden Sie unter AWS -Compliance-Programme.

Payment Card Industry Data Security Standards (PCI DSS)

Es ist wichtig, dass Sie den gesamten Fluss von Karteninhaberdaten (CHD) innerhalb der Umgebung verstehen, wenn Sie sich an PCI DSS halten. Der CHD-Ablauf bestimmt die Anwendbarkeit des PCI DSS, definiert die Grenzen und Komponenten einer Karteninhaberdatenumgebung (CDE) und damit den Umfang einer PCI-DSS-Bewertung. Die genaue Festlegung des PCI-DSS-Umfangs ist entscheidend für die Festlegung der Sicherheitslage und letztlich für eine erfolgreiche Bewertung. Kunden müssen über ein Verfahren zur Festlegung des Umfangs verfügen, das dessen Vollständigkeit sicherstellt und Änderungen oder Abweichungen im Umfang erkennt.

Der temporäre Charakter containerisierter Anwendungen sorgt für zusätzliche Komplexität bei der Prüfung von Konfigurationen. Aus diesem Grund müssen Kunden stets über alle Parameter der Container-Konfiguration informiert sein, um sicherzustellen, dass die Compliance-Anforderungen in allen Phasen des Container-Lebenszyklus erfüllt werden.

Weitere Informationen zum Erreichen der PCI-DSS-Konformität mit Amazon ECS finden Sie in den folgenden Whitepapers.

- Architektur auf Amazon ECS für PCI-DSS-Konformität
- Architektur für PCI-DSS-Skalierung und -Segmentierung auf AWS

HIPAA (U.S. Health Insurance Portability and Accountability Act)

Die Verwendung von Amazon ECS mit Workloads, die geschützte Gesundheitsinformationen (PHI) verarbeiten, erfordert keine zusätzliche Konfiguration. Amazon ECS fungiert als Orchestrierungsservice, der den Start von Containern auf Amazon EC2 koordiniert. Er arbeitet nicht mit oder auf Daten innerhalb des zu orchestrierenden Workloads. Gemäß den HIPAA-Vorschriften und dem AWS Business Associate Addendum sollte PHI während der Übertragung und im Ruhezustand verschlüsselt werden, wenn Container, die mit Amazon ECS gestartet wurden, darauf zugreifen.

Für jede AWS Speicheroption sind verschiedene Mechanismen zur Verschlüsselung im Ruhezustand verfügbar, z. B. Amazon S3, Amazon EBS und. AWS KMS Sie können ein Overlay-Netzwerk (wie VNS3 oder Weave Net) einsetzen, um eine vollständige Verschlüsselung der zwischen Containern übertragenen PHI-Daten sicherzustellen oder um eine redundante Verschlüsselungsebene bereitzustellen. Die vollständige Protokollierung sollte ebenfalls aktiviert sein und alle Container-Protokolle sollten an Amazon weitergeleitet werden CloudWatch. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

AWS Security Hub

Wird verwendet AWS Security Hub , um Ihre Nutzung von Amazon ECS in Bezug auf bewährte Sicherheitsmethoden zu überwachen. Security Hub verwendet Kontrollen für die Bewertung von Ressourcenkonfigurationen und Sicherheitsstandards, um Sie bei der Einhaltung verschiedener Compliance-Frameworks zu unterstützen. Weitere Informationen zur Verwendung von Security Hub zur Bewertung von Amazon-ECS-Ressourcen finden Sie unter Amazon-ECS-Steuerelemente im AWS Security Hub -Benutzerhandbuch.

Amazon GuardDuty mit Amazon ECS-Laufzeitüberwachung

Amazon GuardDuty ist ein Service zur Bedrohungserkennung, der Ihnen hilft, Ihre Konten, Container, Workloads und die Daten in Ihrer AWS Umgebung zu schützen. Mithilfe von Modellen für maschinelles Lernen (ML) und Funktionen zur Erkennung von Anomalien und Bedrohungen werden GuardDuty kontinuierlich verschiedene Protokollquellen und Laufzeitaktivitäten überwacht, um potenzielle Sicherheitsrisiken und böswillige Aktivitäten in Ihrer Umgebung zu identifizieren und zu priorisieren.

Verwenden Sie Runtime Monitoring GuardDuty , um bösartiges oder nicht autorisiertes Verhalten zu identifizieren. Runtime Monitoring schützt Workloads, die auf Fargate und EC2 ausgeführt werden,

indem es die AWS Protokoll- und Netzwerkaktivitäten kontinuierlich überwacht, um böswilliges oder nicht autorisiertes Verhalten zu identifizieren. Runtime Monitoring verwendet einen schlanken, vollständig verwalteten GuardDuty Security Agent, der das Verhalten auf dem Host analysiert, z. B. den Dateizugriff, die Prozessausführung und Netzwerkverbindungen. Dies deckt Probleme wie die Eskalation von Rechten, die Verwendung offengelegter Anmeldeinformationen oder die Kommunikation mit bösartigen IP-Adressen, Domänen und das Vorhandensein von Malware auf Ihren Amazon EC2 EC2-Instances und Container-Workloads ab. Weitere Informationen finden Sie unter GuardDuty Runtime Monitoring im GuardDuty Benutzerhandbuch.

Empfehlungen zur Einhaltung von Vorschriften

Sie sollten die Eigentümer der Compliance-Programme in Ihrem Unternehmen frühzeitig einbinden und das AWS -Modell der geteilten Verantwortung nutzen, um die Verantwortlichkeiten für die Compliance-Kontrolle festzulegen, damit die entsprechenden Compliance-Programme erfolgreich durchgeführt werden können.

AWS Fargate Bundesstandard für Informationsverarbeitung (FIPS-140)

Bundesstandard für Informationsprozesse (FIPS). FIPS-140 ist ein US-amerikanischer und kanadischer Regierungsstandard, mit dem die Sicherheitsanforderungen für Verschlüsselungsmodule angegeben werden, die vertrauliche Informationen schützen. FIPS-140 definiert eine Reihe validierter Kryptografiefunktionen, mit denen Daten während der Übertragung und Daten im Ruhezustand verschlüsselt werden können.

Wenn Sie die FIPS-140-Konformität aktivieren, können Sie Workloads auf Fargate auf FIPS-140-konforme Weise ausführen. Weitere Informationen über FIPS-140-Konformität finden Sie unter Bundesstandard für Informationsprozesse (FIPS) 140-2.

AWS Fargate Überlegungen zu FIPS-140

Beachten Sie die folgenden Punkte, wenn Sie die FIPS-140-Konformität auf Fargate nutzen:

- Die FIPS-140-Konformität ist nur in den Regionen AWS GovCloud (US) verfügbar.
- Die FIPS-140-Konformität ist standardmäßig deaktiviert. Sie müssen sie einschalten.
- Ihre Aufgaben müssen die folgende Konfiguration verwenden, um die FIPS-140-Konformität zu gewährleisten:
  - Der `operatingSystemFamily` muss `LINUX` sein.

- Der `cpuArchitecture` muss `X86_64` sein.
- Die Fargate-Plattformversion muss `1.4.0` oder höher sein.

FIPS auf Fargate verwenden

Gehen Sie wie folgt vor, um die FIPS-140-Konformität auf Fargate zu nutzen.

1. Schalten Sie die FIPS-140-Konformität ein. Weitere Informationen finden Sie unter the section called “AWS Fargate Einhaltung des Federal Information Processing Standard (FIPS-140)”.
2. Sie können optional ECS Exec verwenden, um den folgenden Befehl auszuführen, um den FIPS-140-Konformitätsstatus für einen Cluster zu überprüfen.

   Ersetzen Sie *my-cluster* durch den Namen Ihres Clusters.

   Ein Rückgabewert von „1“ gibt an, dass Sie FIPS verwenden.

   ```
   aws ecs execute-command --cluster cluster-name \
       --interactive \
       --command "cat /proc/sys/crypto/fips_enabled"
   ```

Verwendung CloudTrail für Fargate FIPS-140-Audits

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn API- und Konsolenaktivitäten in Amazon ECS auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto ansehen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung von Ereignissen in Ihrem AWS Konto, einschließlich Ereignissen für Amazon ECS, erstellen Sie einen Trail, der zur Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket CloudTrail verwendet wird. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie unter the section called “Amazon ECS-API-Aufrufe protokollieren mit AWS CloudTrail”.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `PutAccountSettingDefault` API-Aktion demonstriert:


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIV5AJI5LXF5EXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIPWIOFC3EXAMPLE",
    },
    "eventTime": "2023-03-01T21:45:18Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "PutAccountSettingDefault",
    "awsRegion": "us-gov-east-1",
    "sourceIPAddress": "52.94.133.131",
    "userAgent": "aws-cli/2.9.8 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ecs.put-account-setting",
    "requestParameters": {
        "name": "fargateFIPSMode",
        "value": "enabled"
    },
    "responseElements": {
        "setting": {
            "name": "fargateFIPSMode",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/jdoe"
        }
    },
    "requestID": "acdc731e-e506-447c-965d-f5f75EXAMPLE",
    "eventID": "6afced68-75cd-4d44-8076-0beEXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ecs-fips.us-gov-east-1.amazonaws.com"
    }
}
```

Infrastruktursicherheit in Amazon Elastic Container Service

Als verwalteter Service ist Amazon Elastic Container Service durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon ECS zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können diese API-Vorgänge von einem beliebigen Netzwerkstandort aus aufrufen. Amazon ECS unterstützt ressourcenbasierte Zugriffsrichtlinien, die Einschränkungen auf der Grundlage der Quell-IP-Adresse enthalten können. Stellen Sie daher sicher, dass die Richtlinien die IP-Adresse für den Netzwerkstandort berücksichtigen. Sie können auch Amazon-ECS-Richtlinien verwenden, um den Zugriff über bestimmte Amazon Virtual Private Cloud-Endpunkte oder bestimmte VPCs zu steuern. Dadurch wird der Netzwerkzugriff auf eine bestimmte Amazon ECS-Ressource effektiv nur von der spezifischen VPC innerhalb des AWS Netzwerks isoliert. Weitere Informationen finden Sie unter Amazon ECS und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).

Amazon ECS und Schnittstellen-VPC-Endpunkte (AWS PrivateLink)

Sie können die Sicherheit Ihrer VPC erhöhen, indem Sie Amazon ECS so konfigurieren, dass ein Schnittstellen-VPC-Endpunkt verwendet wird. Schnittstellenendpunkte werden von einer Technologie unterstützt AWS PrivateLink, mit der Sie privat auf Amazon ECS-APIs zugreifen können, indem Sie private IP-Adressen verwenden. AWS PrivateLink schränkt den gesamten Netzwerkverkehr zwischen Ihrer VPC und Amazon ECS auf das Amazon-Netzwerk ein. Sie benötigen kein Internet-Gateway, kein NAT-Gerät und kein Virtual Private Gateway.

Weitere Informationen zu AWS PrivateLink VPC-Endpunkten finden Sie unter VPC-Endpunkte im Amazon VPC-Benutzerhandbuch.

Überlegungen

Überlegungen zu Endpunkten in Regionen wurden ab dem 23. Dezember 2023 eingeführt

Seien Sie sich der folgenden Aspekte bewusst, bevor Sie VPC-Endpunkte für Amazon ECS einrichten:

- Sie müssen über die folgenden regionsspezifischen VPC-Endpunkte verfügen:
  - `com.amazonaws.region.ecs-agent`
  - `com.amazonaws.region.ecs-telemetry`
  - `com.amazonaws.region.ecs`

  Beispielsweise benötigt die Region Kanada West (Calgary) (ca-west-1) die folgenden VPC-Endpunkte:
  - `com.amazonaws.ca-west-1.ecs-agent`
  - `com.amazonaws.ca-west-1.ecs-telemetry`
  - `com.amazonaws.ca-west-1.ecs`
- Wenn Sie eine Vorlage verwenden, um AWS Ressourcen in der neuen Region zu erstellen, und die Vorlage aus einer Region kopiert wurde, die vor dem 23. Dezember 2023 eingeführt wurde, führen Sie je nach Region, aus der kopiert wurde, einen der folgenden Vorgänge aus.

  Die Region, aus der kopiert wurde, ist beispielsweise USA Ost (Nord-Virginia) (us-east-1). Die Copy-to-Region ist Canada West (Calgary) (ca-west-1).

Konfiguration	Aktion	
Die Region, aus der kopiert wurde, hat keine VPC-Endpunkte.	Erstellen Sie alle drei VPC-Endpoints für die neue Region (z. B. `com.amazonaws.ca-west-1.ecs-agent`).	

Konfiguration	Aktion	
Die kopierte Region enthält regionsspezifische VPC-Endpunkte.	a. Erstellen Sie alle drei VPC-Endpoints für die neue Region (z. B.`com.amazonaws.ca-west-1.ecs-agent`). b. Löschen Sie alle drei VPC-Endpoints für die Copy-From-Region (z. B.). `com.amazonaws.us-east-1.ecs-agent`	

Überlegungen zu Amazon-ECS-VPC-Endpunkten für den Starttyp Fargate

Wenn es einen VPC-Endpunkt für `ecr.dkr` und `ecr.api` in derselben VPC gibt, in der eine Fargate-Aufgabe bereitgestellt wird, verwendet sie den VPC-Endpunkt. Wenn es keinen VPC-Endpunkt gibt, wird die Fargate-Schnittstelle verwendet.

Seien Sie sich der folgenden Aspekte bewusst, bevor Sie VPC-Endpunkte für Amazon ECS einrichten:

- Aufgaben, die den Starttyp Fargate verwenden, benötigen nicht die Schnittstellen-VPC-Endpunkte für Amazon ECS. Möglicherweise benötigen Sie jedoch Schnittstellen-VPC-Endpunkte für Amazon ECR, Secrets Manager oder Amazon CloudWatch Logs, die in den folgenden Punkten beschrieben werden.
  - Damit Ihre Aufgaben private Images von Amazon ECR abrufen können, müssen Sie Schnittstellen-VPC-Endpunkte für Amazon ECR erstellen. Weitere Informationen finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon Elastic Container-Registry-Benutzerhandbuch.

    Wenn Ihre VPC kein Internet-Gateway hat, müssen Sie den Gateway-Endpunkt für Amazon S3 erstellen. Weitere Informationen finden Sie unter Erstellen des Amazon-S3-Gateway-Endpunkts im Benutzerhandbuch für Amazon Elastic Container Registry. Die Schnittstellen-Endpunkte für Amazon S3 können nicht mit Amazon ECR verwendet werden.

> **Important**
>
> Wenn Sie Amazon ECR für die Verwendung eines Schnittstellen-VPC-Endpunkts konfigurieren, können Sie eine Aufgabenausführungsrolle erstellen, die die Bedingungsschlüssel für die Einschränkung des Zugriffs auf eine bestimmte VPC oder einen bestimmten VPC-Endpunkt enthält. Weitere Informationen finden Sie unter Fargate-Aufgaben: Abrufen von Amazon ECR-Images über die Berechtigungen von Schnittstellen-Endpunkten.

  - Damit Ihre Aufgaben sensitive Daten von Secrets Manager abrufen können, müssen Sie die Schnittstellen-VPC-Endpunkte für Secrets Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Secrets Manager mit VPC-Endpunkten im AWS Secrets Manager - Benutzerhandbuch.
  - Wenn Ihre VPC kein Internet-Gateway hat und Ihre Aufgaben den `awslogs` Protokolltreiber verwenden, um CloudWatch Protokollinformationen an Logs zu senden, müssen Sie einen VPC-Schnittstellen-Endpunkt für CloudWatch Logs erstellen. Weitere Informationen finden Sie unter Using CloudWatch Logs with Interface VPC Endpoints im Amazon CloudWatch Logs-Benutzerhandbuch.
- VPC-Endpunkte unterstützen derzeit keine regionsübergreifenden Anforderungen. Stellen Sie sicher, dass Sie Ihren Endpunkt innerhalb derselben Region erstellen, in der Sie Ihre API-Aufrufe an Amazon ECS ausgeben möchten. Nehmen wir zum Beispiel an, Sie möchten Aufgaben in USA Ost (Nord-Virginia) ausführen. Anschließend müssen Sie den Amazon-ECS-VPC-Endpunkt in USA Ost (Nord-Virginia) erstellen. Ein in einer anderen Region erstellter Amazon-ECS-VPC-Endpunkt kann keine Aufgaben in USA Ost (Nord-Virginia) ausführen.
- VPC-Endpunkte unterstützen nur von Amazon bereitgestellten DNS über Amazon Route 53. Wenn Sie Ihre eigene DNS verwenden möchten, können Sie die bedingte DNS-Weiterleitung nutzen. Weitere Informationen finden Sie unter DHCP Options Sets im Amazon VPC-Benutzerhandbuch.
- Die Sicherheitsgruppe für den VPC-Endpunkt müssen eingehende Verbindungen auf TCP-Port 443 aus dem privaten Subnetz der VPC zulassen.
- Die Service-Connect-Verwaltung des Envoy-Proxys verwendet den `com.amazonaws.`*`region`*`.ecs-agent`-VPC-Endpunkt. Wenn Sie die VPC-Endpunkte nicht verwenden, verwendet die Service-Connect-Verwaltung des Envoy-Proxys den `ecs-sc`-Endpunkt in dieser Region. Eine Liste der Amazon-ECS-Endpunkte in jeder Region finden Sie unter Amazon-ECS-Endpunkte und Kontingente.

Überlegungen zu Amazon-ECS-VPC-Endpunkten für den Starttyp EC2

Seien Sie sich der folgenden Aspekte bewusst, bevor Sie VPC-Endpunkte für Amazon ECS einrichten:

- Für Aufgaben, die den EC2-Starttyp verwenden, müssen die Container-Instances, für die sie gestartet werden, mindestens Version `1.25.1` oder höher des Amazon-ECS-Container-Agents ausführen. Weitere Informationen finden Sie unter Verwaltung von Amazon ECS Linux-Container-Instances.
- Damit Ihre Aufgaben sensitive Daten von Secrets Manager abrufen können, müssen Sie die Schnittstellen-VPC-Endpunkte für Secrets Manager erstellen. Weitere Informationen finden Sie unter Verwenden von Secrets Manager mit VPC-Endpunkten im AWS Secrets Manager - Benutzerhandbuch.
- Wenn Ihre VPC kein Internet-Gateway hat und Ihre Aufgaben den `awslogs` Protokolltreiber verwenden, um CloudWatch Protokollinformationen an Logs zu senden, müssen Sie einen VPC-Schnittstellen-Endpunkt für CloudWatch Logs erstellen. Weitere Informationen finden Sie unter Using CloudWatch Logs with Interface VPC Endpoints im Amazon CloudWatch Logs-Benutzerhandbuch.
- VPC-Endpunkte unterstützen derzeit keine regionsübergreifenden Anforderungen. Stellen Sie sicher, dass Sie Ihren Endpunkt innerhalb derselben Region erstellen, in der Sie Ihre API-Aufrufe an Amazon ECS ausgeben möchten. Nehmen wir zum Beispiel an, Sie möchten Aufgaben in USA Ost (Nord-Virginia) ausführen. Anschließend müssen Sie den Amazon-ECS-VPC-Endpunkt in USA Ost (Nord-Virginia) erstellen. Ein Amazon ECS-VPC-Endpunkt, der in einer anderen Region erstellt wurde, kann keine Aufgaben in USA Ost (Nord-Virginia) ausführen.
- VPC-Endpunkte unterstützen nur von Amazon bereitgestellten DNS über Amazon Route 53. Wenn Sie Ihre eigene DNS verwenden möchten, können Sie die bedingte DNS-Weiterleitung nutzen. Weitere Informationen finden Sie unter DHCP Options Sets im Amazon VPC-Benutzerhandbuch.
- Die Sicherheitsgruppe für den VPC-Endpunkt müssen eingehende Verbindungen auf TCP-Port 443 aus dem privaten Subnetz der VPC zulassen.

Erstellen der VPC-Endpunkte für Amazon ECS

Um den VPC-Endpunkt für den Amazon-ECS-Service zu erstellen, verwenden Sie die Prozedur Creating an Interface Endpoint (Einen Schnittstellen-Endpunkt erstellen) im Amazon VPC-Benutzerhandbuch, um die folgenden Endpunkte zu erstellen: Wenn in Ihrer VPC Container-Instances vorhanden sind, sollten Sie die Endpunkte in der Reihenfolge erstellen, in der sie

aufgelistet werden. Wenn Sie vorhaben, Ihre Container-Instances nach Ihrem VPC-Endpunkt zu erstellen, dann spielt die Reihenfolge keine Rolle.

- `com.amazonaws.region.ecs-agent`
- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

> **Note**
>
> *region* repräsentiert die Regions-ID für eine von AWS unterstützte Amazon-ECS-Region, z. B. `us-east-2` für die Region USA Ost (Ohio).

Der `ecs-agent` Endpunkt verwendet die `ecs:poll` API und der `ecs-telemetry` Endpunkt verwendet die `ecs:poll` `ecs:StartTelemetrySession` AND-API.

Wenn Aufgaben vorhanden sind, die den EC2-Starttyp verwenden, muss jede Container-Instance nach dem Erstellen der VPC-Endpunkte die neue Konfiguration übernehmen. Damit dies möglich ist, müssen Sie entweder jede Container-Instance neu starten oder den Amazon-ECS-Container-Agenten für jede Container-Instance neu starten. Um den Container-Agenten neu zu starten, führen Sie die folgenden Schritte aus.

Neustarten des Amazon-ECS-Container-Agenten

1. Melden Sie sich bei Ihrer Container-Instance über SSH an.
2. Halten Sie den Container-Agent an.

   ```
   sudo docker stop ecs-agent
   ```

3. Starten Sie den Container-Agenten.

   ```
   sudo docker start ecs-agent
   ```

Nachdem Sie die VPC-Endpunkte erstellt haben und der Amazon-ECS-Container-Agent für jede Container-Instance neu gestartet wurde, wird die neue Konfiguration von allen neu gestarteten Aufgaben übernommen.

Erstellen einer VPC-Endpunktrichtlinie für Amazon ECS

Sie können eine Endpunktrichtlinie an Ihren VPC-Endpunkt anhängen, der den Zugriff auf Amazon ECS steuert. Die Richtlinie gibt die folgenden Informationen an:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Beispiel: VPC-Endpunktrichtlinie für Amazon-ECS-Aktionen

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für Amazon ECS. Wenn diese Richtlinie an einen Endpunkt angefügt wird, gewährt sie Zugriff auf die Berechtigung zum Erstellen und Auflisten von Clustern. Die Aktionen `CreateCluster` und `ListClusters` akzeptieren keine Ressourcen, daher wird die Ressourcendefinition auf * für alle Ressourcen festgelegt.

```
{
   "Statement":[
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Bewährte Methoden zur Aufgaben- und Containersicherheit von Amazon ECS

Sie sollten das Container-Image als erste Verteidigungslinie gegen einen Angriff betrachten. Ein unsicheres, schlecht aufgebautes Image kann es einem Angreifer ermöglichen, die Grenzen des

Containers zu umgehen und sich Zugriff auf den Host zu verschaffen. Sie sollten wie folgt vorgehen, um das Risiko eines solchen Vorfalls zu verringern.

Wir empfehlen Folgendes, wenn Sie Ihre Aufgaben und Container einrichten.

Minimale Images erstellen oder distroless-Images verwenden

Entfernen Sie zunächst alle überflüssigen Binärdateien aus dem Container-Image. Wenn Sie ein unbekanntes Image aus Amazon ECR Public Gallery verwenden, überprüfen Sie das Image, um auf den Inhalt der einzelnen Ebenen des Containers hinzuweisen. Sie können dafür eine Anwendung wie Dive verwenden.

Alternativ können Sie Images distroless verwenden, die nur Ihre Anwendung und ihre Laufzeitabhängigkeiten enthalten. Sie enthalten keine Paketmanager oder Shells. Distroless Images verbessern das „Signal-Rausch-Verhältnis von Scannern und reduzieren den Aufwand für den Nachweis der Herkunft auf das, was Sie brauchen.“ Weitere Informationen finden Sie in der GitHub Dokumentation zu distroless.

Docker verfügt über einen Mechanismus zum Erstellen von Images aus einem reservierten, minimalen Image, das als Scratch bezeichnet wird. Weitere Informationen finden Sie in der Docker-Dokumentation unter Erstellen eines einfachen übergeordneten Images mithilfe von Scratch. Mit Sprachen wie Go können Sie eine statische verknüpfte Binärdatei erstellen und in Ihrem Dockerfile darauf verweisen. Das folgende Beispiel zeigt, wie Sie dies erreichen können.

```
############################
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
```

```
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

Das vorherige Beispiel ist auch ein Beispiel für einen mehrstufigen Build. Diese Art von Builds ist vom Standpunkt der Sicherheit aus attraktiv, da Sie damit die Größe des endgültigen Images, das in Ihre Container-Registrierung übertragen wird, minimieren können. Container-Images ohne Build-Tools und andere überflüssige Binärdateien verbessern Ihre Sicherheitslage, indem sie die Angriffsfläche des Images reduzieren. Weitere Informationen zu mehrstufigen Builds finden Sie unter Erstellen mehrstufiger Builds.

Ihre Images auf Schwachstellen scannen

Ähnlich wie ihre Pendants in virtuellen Maschinen können Container-Images Binärdateien und Anwendungsbibliotheken mit Schwachstellen enthalten oder mit der Zeit Schwachstellen entwickeln. Am besten schützen Sie sich vor Angriffen, indem Sie Ihre Images regelmäßig mit einem Image-Scanner überprüfen.

Images, die in Amazon ECR gespeichert sind, können per Push oder auf Abruf (einmal alle 24 Stunden) gescannt werden. Amazon ECR Basic Scanning verwendet Clair, eine Open-Source-Lösung zum Scannen von Images. Das erweiterte Scannen mit Amazon ECR verwendet Amazon Inspector. Nachdem ein Bild gescannt wurde, werden die Ergebnisse im Amazon ECR-Event-Stream in Amazon EventBridge protokolliert. Sie können die Ergebnisse eines Scans auch in der Amazon ECR-Konsole oder durch Aufrufen der DescribeImageScanFindingsAPI anzeigen. Images mit einer `HIGH`- oder `CRITICAL`-Schwachstelle sollten gelöscht oder neu erstellt werden. Wenn ein bereitgestelltes Image eine Schwachstelle aufweist, sollte es so schnell wie möglich ersetzt werden.

Docker Desktop Edge Version 2.3.6.0 oder höher kann lokale Images scannen. Die Scans werden von Snyk, einem Anwendungssicherheitsservice, unterstützt. Wenn Schwachstellen entdeckt werden, identifiziert Snyk die Ebenen und Abhängigkeiten mit der Sicherheitslücke im Dockerfile. Es empfiehlt auch sichere Alternativen wie die Verwendung eines schlankeren Basis-Images mit weniger Schwachstellen oder die Aktualisierung eines bestimmten Pakets auf eine neuere Version. Mithilfe von Docker-Scan können Entwickler potenzielle Sicherheitsprobleme lösen, bevor sie ihre Images in die Registrierung übertragen.

- Automatisieren der Image-Compliance mithilfe von Amazon ECR und AWS Security Hub erklärt, wie Informationen zu Oberflächen-Schwachstellen aus Amazon ECR in AWS Security Hub abgerufen und deren Behebung automatisiert werden können, indem der Zugriff auf anfällige Images blockiert wird.

Sonderberechtigungen aus Ihren Images entfernen

Die Flags für Zugriffsrechte `setuid` und `setgid` erlauben die Ausführung einer ausführbaren Datei mit den Berechtigungen des Eigentümers oder der Gruppe der ausführbaren Datei. Entfernen Sie alle Binärdateien mit diesen Zugriffsrechten aus Ihrem Image, da diese Binärdateien zur Erweiterung von Rechten verwendet werden können. Erwägen Sie, alle Shells und Serviceprogramme wie `nc` und `curl`, die für böswillige Zwecke verwendet werden können, zu entfernen. Sie können die Dateien mit `setuid`- und `setgid`-Zugriffsrechten finden, indem Sie den folgenden Befehl verwenden.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Um diese speziellen Berechtigungen aus diesen Dateien zu entfernen, fügen Sie Ihrem Container-Image die folgende Direktive hinzu.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Eine Reihe von kuratierten Images erstellen

Anstatt es Entwicklern zu ermöglichen, ihre eigenen Images zu erstellen, sollten Sie eine Reihe von geprüften Images für die verschiedenen Anwendungsstapel in Ihrem Unternehmen erstellen. Auf diese Weise können Entwickler darauf verzichten, zu lernen, wie man Dockerfiles erstellt, und sich auf das Schreiben von Code konzentrieren. Wenn Änderungen in Ihrer Codebasis zusammengeführt werden, kann eine CI/CD-Pipeline das Asset automatisch kompilieren und dann in einem Artefakt-Repository speichern. Und zuletzt kopieren Sie das Artefakt in das entsprechende Image, bevor Sie es in eine Docker-Registry wie Amazon ECR übertragen. Zumindest sollten Sie eine Reihe von Basis-Images erstellen, aus denen Entwickler ihre eigenen Dockerfiles erstellen können. Sie sollten es vermeiden, Bilder aus Docker Hub abzurufen. Sie wissen nicht immer, was sich im Image befindet, und etwa ein Fünftel der Top-1000 Images weist Schwachstellen auf. Eine Liste dieser Images und ihrer Schwachstellen finden Sie unter https://vulnerablecontainers.org/.

Anwendungspakete und Bibliotheken auf Schwachstellen scannen

Die Verwendung von Open-Source-Bibliotheken ist heute weit verbreitet. Wie bei Betriebssystemen und Betriebssystempaketen können diese Bibliotheken Schwachstellen aufweisen. Im Rahmen des Entwicklungszyklus sollten diese Bibliotheken gescannt und aktualisiert werden, wenn kritische Schwachstellen gefunden werden.

Docker Desktop führt lokale Scans mit Snyk durch. Es kann auch verwendet werden, um Schwachstellen und potenzielle Lizenzprobleme in Open-Source-Bibliotheken zu finden. Es kann direkt in Entwickler-Workflows integriert werden, sodass Sie die von Open-Source-Bibliotheken ausgehenden Risiken minimieren können. Weitere Informationen finden Sie unter den folgenden Themen:

- Die Open Source Application Security Tools enthalten eine Liste von Tools zur Erkennung von Schwachstellen in Anwendungen.

Eine statische Codeanalyse durchführen

Sie sollten eine statische Codeanalyse durchführen, bevor Sie ein Container-Image erstellen. Sie wird anhand Ihres Quellcodes durchgeführt und dient dazu, Codierungsfehler und Code zu identifizieren, der von böswilligen Akteuren ausgenutzt werden könnte, z. B. bei Fehlerinjektionen. SonarQubeist eine beliebte Option für statische Anwendungssicherheitstests (SAST) mit Unterstützung für eine Vielzahl verschiedener Programmiersprachen.

Container als nicht Root-Benutzer ausführen

Sie sollten Container als Nicht-Root-Benutzer ausführen. Standardmäßig werden Container als der `root`-Benutzer ausgeführt, sofern die `USER`-Direktive nicht in Ihrem Dockerfile enthalten ist. Die standardmäßigen Linux-Funktionen, die von Docker zugewiesen werden, schränken die Aktionen ein, die als `root` ausgeführt werden können, aber nur geringfügig. Beispielsweise darf ein Container, der als `root` ausgeführt wird, immer noch nicht auf Geräte zugreifen.

Als Teil Ihrer CI/CD-Pipeline sollten Sie Dockerfiles so einrichten, dass es nach der `USER`-Direktive sucht und den Build fehlschlägt, falls sie fehlt. Weitere Informationen finden Sie unter den folgenden Themen:

- DockerFile-Lint ist ein Open-Source-Tool von RedHat , mit dem überprüft werden kann, ob die Datei den Best Practices entspricht.

- Hadolint ist ein weiteres Tool zum Erstellen von Docker-Images, die den bewährten Methoden entsprechen.

Ein schreibgeschütztes Root-Dateisystem verwenden

Sie sollten ein schreibgeschütztes Root-Dateisystem verwenden. Das Root-Dateisystem eines Containers ist standardmäßig beschreibbar. Wenn Sie einen Container mit einem `RO` (schreibgeschützten) Root-Dateisystem konfigurieren, müssen Sie explizit definieren, wo Daten persistent gespeichert werden können. Dadurch wird Ihre Angriffsfläche reduziert, da in das Dateisystem des Containers nur geschrieben werden kann, wenn ausdrückliche Berechtigungen erteilt wurden.

> **Note**
>
> Ein schreibgeschütztes Root-Dateisystem kann zu Problemen mit bestimmten Betriebssystempaketen führen, die erwarten, in das Dateisystem schreiben zu können. Wenn Sie vorhaben, schreibgeschützte Root-Dateisysteme zu verwenden, testen Sie es vorher gründlich.

Aufgaben mit CPU- und Speicherlimits konfigurieren (Amazon EC2)

Sie sollten Aufgaben mit CPU- und Speicherlimits konfigurieren, um das folgende Risiko zu minimieren. Die Ressourcenlimits einer Aufgabe legen eine Obergrenze für die Menge an CPU und Arbeitsspeicher fest, die von allen Containern innerhalb einer Aufgabe reserviert werden kann. Wenn keine Grenzwerte festgelegt sind, haben Aufgaben Zugriff auf die CPU und den Arbeitsspeicher des Hosts. Dies kann zu Problemen führen, bei denen Aufgaben, die auf einem gemeinsam genutzten Host bereitgestellt werden, anderen Aufgaben die Systemressourcen entziehen können.

> **Note**
>
> Bei Amazon ECS für AWS Fargate Aufgaben müssen Sie CPU- und Speicherlimits angeben, da diese Werte für Abrechnungszwecke verwendet werden. Eine Aufgabe, die alle Systemressourcen beansprucht, ist für Amazon ECS Fargate kein Problem, da jede Aufgabe auf einer eigenen Dedicated Instance ausgeführt wird. Wenn Sie kein Speicherlimit angeben, weist Amazon ECS jedem Container mindestens 4 MB zu. Wenn für die Aufgabe kein CPU-

Limit festgelegt ist, weist ihr der Amazon-ECS-Container-Agent ebenfalls mindestens 2 CPUs zu.

Unveränderliche Tags mit Amazon ECR verwenden

Mit Amazon ECR können und sollten Sie konfigurierte Images mit unveränderlichen Tags verwenden. Dadurch wird verhindert, dass eine geänderte oder aktualisierte Version eines Images mit einem identischen Tag in Ihr Image-Repository übertragen wird. Dies schützt davor, dass ein Angreifer eine kompromittierte Version eines Images über Ihr Image mit demselben Tag überträgt. Durch die Verwendung unveränderlicher Tags zwingen Sie sich quasi dazu, bei jeder Änderung ein neues Image mit einem anderen Tag zu übertragen.

Vermeiden, Container als privilegiert auszuführen (Amazon EC2)

Sie sollten es vermeiden, Container als privilegiert auszuführen. Für den Hintergrund werden Container als `privileged` mit erweiterten Rechten auf dem Host ausgeführt. Das bedeutet, dass der Container alle Linux-Funktionen erbt, die `root` auf dem Host zugewiesen sind. Seine Verwendung sollte stark eingeschränkt oder verboten werden. Wir empfehlen, die Amazon-ECS-Umgebungsvariable `ECS_DISABLE_PRIVILEGED` für den Container-Agenten auf `true` einzustellen, sodass Container nicht `privileged` auf bestimmten Hosts ausführen, wenn `privileged` nicht benötigt ist. Alternativ können Sie Ihre Aufgabendefinitionen AWS Lambda auf die Verwendung des `privileged` Parameters überprüfen.

> **Note**
>
> Das Ausführen eines Containers als `privileged` wird auf Amazon ECS auf AWS Fargate nicht unterstützt.

Nicht benötigte Linux-Funktionen aus dem Container entfernen

Im Folgenden finden Sie eine Liste der Linux-Standardfunktionen, die Docker-Containern zugewiesen sind. Weitere Informationen zu den einzelnen Funktionen finden Sie unter Linux-Funktionen im Überblick.

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
```

```
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Wenn ein Container nicht alle der oben aufgeführten Docker-Kernelfunktionen benötigt, sollten Sie erwägen, sie aus dem Container zu löschen. Weitere Informationen zu den einzelnen Docker-Kernel-Fähigkeiten finden Sie unter KernelCapabilities. Sie können herausfinden, welche Funktionen verwendet werden, indem Sie wie folgt vorgehen:

- Installieren Sie das Betriebssystempaket libcap-ng und führen Sie das `pscap`-Hilfsprogramm aus, um die Funktionen aufzulisten, die jeder Prozess verwendet.
- Sie können Capsh auch verwenden, um zu entziffern, welche Fähigkeiten ein Prozess verwendet.

Verwenden Sie einen kundenseitig verwalteten Schlüssel (CMK), um Images zu verschlüsseln, die an Amazon ECR übertragen werden

Sie sollten einen kundenseitig verwalteten Schlüssel (CMK) verwenden, um Images zu verschlüsseln, die an Amazon ECR übertragen werden. Bilder, die an Amazon ECR übertragen werden, werden im Ruhezustand automatisch mit einem AWS Key Management Service (AWS KMS) verwalteten Schlüssel verschlüsselt. Wenn Sie lieber Ihren eigenen Schlüssel verwenden möchten, unterstützt Amazon ECR jetzt die AWS KMS Verschlüsselung mit vom Kunden verwalteten Schlüsseln (CMK). Bevor Sie die serverseitige Verschlüsselung mit einem CMK aktivieren, sollten Sie die in der Dokumentation zur Verschlüsselung im Ruhezustand aufgeführten Überlegungen lesen.

Tutorials für Amazon ECS

Die folgenden Tutorials veranschaulichen, wie Sie mit Amazon ECS gängige Aufgaben durchführen.

Sie können jedes der folgenden Tutorials verwenden, um Aufgaben auf Amazon ECS bereitzustellen, indem Sie AWS CLI

Tutorial-Übersicht	Weitere Informationen
Erstellen Sie eine Linux-Aufgabe für den Fargate-Starttyp.	Erstellen einer Amazon ECS-Linux-Aufgabe für den Fargate-Starttyp mit dem AWS CLI
Erstellen Sie eine Windows-Aufgabe für den Fargate-Starttyp.	Erstellen einer Amazon ECS-Windows-Aufgabe für den Fargate-Starttyp mit dem AWS CLI
Erstellen Sie eine Linux-Aufgabe für den EC2-Starttyp.	Erstellen einer Amazon ECS-Aufgabe für den EC2-Starttyp mit dem AWS CLI

Sie können jedes der folgenden Tutorials verwenden, um mehr über Überwachung und Protokollierung zu erfahren.

Tutorial-Übersicht	Weitere Informationen
Richten Sie eine einfache Lambda-Funktion ein, die auf Aufgabenereignisse wartet und diese in einen CloudWatch Logs-Log-Stream schreibt.	Konfiguration von Amazon ECS für die Überwachung von CloudWatch Ereignissen
Konfigurieren Sie eine EventBridge Amazon-Er	Senden von Amazon Simple Notification Service-B

Tutorial-Übersicht	Weitere Informationen	
eignisregel, die nur Aufgabene reignisse erfasst, bei denen die Ausführung der Aufgabe beendet wurde, weil einer ihrer wichtigsten Container beendet wurde.	enachrichtigungen für Ereignisse, bei denen Amazon ECS-Aufgaben gestoppt wurden	
Verketten Sie Protokoll nachrichten, die ursprünglich zu einem Kontext gehörten, aber auf mehrere Datensätze oder Protokollzeilen aufgeteilt wurden.	Verkettung mehrzeiliger Amazon ECS-Protokollnachr ichten oder Stack-Trace-Protokollnachrichten	
Stellen Sie Fluent Bit-Conta iner auf ihren Windows-I nstances bereit, die in Amazon ECS ausgeführt werden, um die von den Windows-A ufgaben generierten Protokoll e CloudWatch zur zentralen Protokollierung an Amazon zu streamen.	Bereitstellung von Fluent Bit auf Amazon ECS-Windows-Containern	

Sie können jedes der folgenden Tutorials verwenden, um mehr über die Verwendung der Active Directory-Authentifizierung mit einem gruppenverwalteten Servicekonto auf Amazon ECS zu erfahren.

Tutorial-Übersicht	Weitere Informationen	
Verwenden Sie ein Gruppenko nto für verwaltete Dienste mit Linux-Containern auf EC2.	Verwendung gMSA für Linux EC2-Container auf Amazon ECS	

Tutorial-Übersicht	Weitere Informationen
Verwenden Sie das verwaltete Gruppendienstkonto mit Windows-Containern auf EC2.	Erfahren Sie, wie Sie GMSAs für EC2-Windows-Container für Amazon ECS verwenden
Verwenden Sie ein gruppenverwaltetes Dienstkonto mit Linux-Containern auf Fargate.	Wird gMSA für Linux Container auf Fargate verwendet
Erstellen Sie eine Aufgabe, die einen Windows-Container ausführt, der über Anmeldeinformationen für den Zugriff auf Active Directory mit einem domänenlosen Gruppenverwalteten Dienstkonto verfügt.	Verwenden von Amazon ECS-Windows-Containern mit domainless unter gMSA Verwendung von AWS CLI

Erstellen einer Amazon ECS-Linux-Aufgabe für den Fargate-Starttyp mit dem AWS CLI

Mithilfe der folgenden Schritte können Sie über die AWS CLI einen Cluster einrichten, eine Aufgabendefinition registrieren, eine Linux-Aufgabe ausführen sowie andere allgemeine Szenarien in Amazon ECS umsetzen. Verwenden Sie die neueste Version von. AWS CLI Weitere Informationen zur Aktualisierung auf die neueste Version finden Sie unter Installieren der AWS Command Line Interface.

Themen

- Schritt 7: Testen
- Schritt 8: Bereinigen

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen zur Installation oder Aktualisierung von finden Sie AWS CLI unter Installation von AWS Command Line Interface.
- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben eine VPC und die zu verwendende Sicherheitsgruppe erstellt. In diesem Tutorial wird ein Container-Image verwendet, das auf Amazon ECR Public gehostet wird, sodass Ihre Aufgabe über Internetzugang verfügen muss. Um Ihrer Aufgabe eine Route zum Internet zu geben, verwenden Sie eine der folgenden Optionen.
  - Verwenden Sie ein privates Subnetz mit einem NAT-Gateway, das über eine Elastic IP-Adresse verfügt.
  - Verwenden Sie ein öffentliches Subnetz und weisen Sie der Aufgabe eine öffentliche IP-Adresse zu.

  Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.

  Informationen zu Sicherheitsgruppen und -regeln finden Sie unter Standard-Sicherheitsgruppen für Ihre VPCs und Beispielregeln im Benutzerhandbuch zur Amazon Virtual Private Cloud.
- Wenn Sie diesem Tutorial mit einem privaten Subnetz folgen, können Sie Amazon ECS Exec verwenden, um direkt mit Ihrem Container zu interagieren und die Bereitstellung zu testen. Sie müssen eine Aufgaben-IAM-Rolle erstellen, um ECS Exec verwenden zu können. Weitere Informationen zur Aufgaben-IAM-Rolle und zu anderen Voraussetzungen finden Sie unter Verwenden von Amazon ECS Exec zum Debuggen.
- (Optional) AWS CloudShell ist ein Tool, das Kunden eine Befehlszeile bietet, ohne ihre eigene EC2-Instance erstellen zu müssen. Weitere Informationen finden Sie unter Was ist? AWS CloudShell im AWS CloudShell Benutzerhandbuch.

Schritt 1: Erstellen eines Clusters

Ihr Konto erhält standardmäßig ein `default` Cluster.

> **Note**
>
> Wenn Sie den bereitgestellten Cluster `default` verwenden, hat dies den Vorteil, dass Sie in den nachfolgenden Befehlen nicht die Option `--cluster` *`cluster_name`* angeben müssen. Wenn Sie Ihren eigenen, nicht standardmäßigen Cluster erstellen, müssen Sie `--cluster` *`cluster_name`* für jeden Befehl angeben, den Sie mit diesem Cluster verwenden möchten.

Erstellen Sie mit dem folgenden Befehl Ihren eigenen Cluster mit eindeutigem Namen:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Ausgabe:

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "fargate-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Schritt 2: Anmelden einer Linux-Aufgabendefinition

Bevor Sie auf Ihrem ECS-Cluster eine Aufgabe ausführen können, müssen Sie eine Aufgabendefinition registrieren. Aufgabendefinitionen sind Listen zusammengefasster Container. Das folgende Beispiel ist eine einfache Aufgabendefinition, die eine PHP-Web-App mit dem httpd-Container-Image erstellt, das auf Docker Hub gehostet wird. Weitere Informationen zu den verfügbaren Parametern für die Aufgabendefinition finden Sie im Abschnitt Amazon-ECS-Aufgabendefinitionen. Für dieses Tutorial wird `taskRoleArn` nur benötigt, wenn Sie die Aufgabe in einem privaten Subnetz bereitstellen und die Bereitstellung testen möchten. Ersetzen Sie `taskRoleArn` die durch die IAM-Aufgabenrolle, die Sie für die Verwendung von ECS Exec erstellt haben, wie unter Voraussetzungen beschrieben.

```
{
       "family": "sample-fargate",
       "networkMode": "awsvpc",
       "taskRoleArn": "arn:aws:iam::aws_account_id:role/execCommandRole",
       "containerDefinitions": [
           {
               "name": "fargate-app",
               "image": "public.ecr.aws/docker/library/httpd:latest",
               "portMappings": [
                   {
                       "containerPort": 80,
                       "hostPort": 80,
                       "protocol": "tcp"
                   }
               ],
               "essential": true,
               "entryPoint": [
                   "sh",
                   "-c"
               ],
               "command": [
                   "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
               ]
           }
       ],
```

```
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

Speichern Sie die JSON-Datei der Aufgabendefinition als Datei und übergeben Sie sie zusammen mit der `--cli-input-json file://`*`path_to_file.json`*-Option.

Verwendung einer JSON-Datei für Containerdefinitionen:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

Der Befehl register-task-definition gibt nach Abschluss der Registrierung eine Beschreibung der Aufgabendefinition zurück.

Schritt 3: Auflisten der Aufgabendefinitionen

Sie können die Aufgabendefinitionen für Ihr Konto jederzeit mit dem Befehl list-task-definitions auflisten. In der Ausgabe dieses Befehls werden die Werte `family` und `revision` angezeigt, die Sie beim Aufruf von run-task oder start-task gemeinsam verwenden können.

```
aws ecs list-task-definitions
```

Ausgabe:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate:1"
    ]
}
```

Schritt 4: Einen Service erstellen

Nachdem Sie eine Aufgabe für Ihr Konto registriert haben, können Sie einen Service für die registrierte Aufgabe in Ihrem Cluster erstellen. In diesem Beispiel erstellen Sie einen Service mit einer Instance der `sample-fargate:1`-Aufgabendefinition, die in Ihrem Cluster ausgeführt wird. Die Aufgabe erfordert eine Route zum Internet, daher gibt es zwei Möglichkeiten, dies zu erreichen.

Eine Möglichkeit besteht darin, ein privates Subnetz zu verwenden, das mit einem NAT-Gateway mit einer Elastic IP-Adresse in einem öffentlichen Subnetz konfiguriert ist. Eine andere Möglichkeit besteht darin, ein öffentliches Subnetz zu verwenden und Ihrer Aufgabe eine öffentliche IP-Adresse zuzuweisen. Wir stellen beide Beispiele unten zur Verfügung.

Beispiel mit einem privaten Subnetz. Die `enable-execute-command` -Option ist erforderlich, um Amazon ECS Exec zu verwenden.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234]}" --enable-execute-command
```

Beispiel mit einem öffentlichen Subnetz.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234],assignPublicIp=ENABLED}"
```

Der Befehl create-service gibt nach Abschluss der Registrierung eine Beschreibung der Aufgabendefinition zurück.

Schritt 5: Auflisten von Services

Listet die Services für Ihren Cluster auf. Der im vorherigen Abschnitt erstellte Service müsste angezeigt werden. Sie können den Servicenamen oder den vollständigen ARN, die/der von diesem Befehl zurückgegeben wird, später zur Beschreibung des Service verwenden.

```
aws ecs list-services --cluster fargate-cluster
```

Ausgabe:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
    ]
}
```

Schritt 6: Beschreibung des gerade ausgeführten Service

Beschreibt Sie den Service unter Verwendung des zuvor abgerufenen Servicenamen, um weitere Informationen über die Aufgabe zu erhalten.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Wenn dies erfolgreich ist, wird eine Beschreibung der Servicefehler und Services zurückgegeben. Im Abschnitt `services` finden Sie beispielsweise Informationen zu Bereitstellungen, z. B. ob Aufgaben den Status ausgeführt oder ausstehend haben. Möglicherweise finden Sie auch Informationen zur Aufgabendefinition, zur Netzwerkkonfiguration und zu Ereignissen mit Zeitstempeln. Im Abschnitt „Fehler“ finden Sie Informationen zu Fehlern (falls vorhanden), die mit dem Aufruf verbunden sind. Informationen zur Fehlerbehebung finden Sie unter Service-Ereignismeldungen. Weitere Informationen zur Servicebeschreibung finden Sie unter Beschreiben von Services.

```
{
    "services": [
        {
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "ENABLED"
                }
            },
            "launchType": "FARGATE",
            "enableECSManagedTags": false,
            "loadBalancers": [],
            "deploymentController": {
                "type": "ECS"
            },
            "desiredCount": 1,
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
```

```
                "minimumHealthyPercent": 100
            },
            "createdAt": 1692283199.771,
            "schedulingStrategy": "REPLICA",
            "placementConstraints": [],
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "ENABLED"
                        }
                    },
                    "pendingCount": 0,
                    "launchType": "FARGATE",
                    "createdAt": 1692283199.771,
                    "desiredCount": 1,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate:1",
                    "updatedAt": 1692283199.771,
                    "platformVersion": "1.4.0",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "serviceName": "fargate-service",
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
```

```
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "status": "ACTIVE",
            "serviceRegistries": [],
            "pendingCount": 0,
            "createdBy": "arn:aws:iam::aws_account_id:user/user_name",
            "platformVersion": "LATEST",
            "placementStrategy": [],
            "propagateTags": "NONE",
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate:1"
        }
    ],
    "failures": []
}
```

Schritt 7: Testen

Die Testaufgabe wurde über ein öffentliches Subnetz bereitgestellt

Beschreiben Sie die Aufgabe im Service, damit Sie die Elastic-Network-Schnittstelle (ENI) für die Aufgabe abrufen können.

Rufen Sie zunächst den ARN der Aufgabe ab.

```
aws ecs list-tasks --cluster fargate-cluster --service fargate-service
```

Die Ausgabe enthält den ARN der Aufgabe.

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE
    ]
```

```
}
```

Beschreiben Sie die Aufgabe und suchen Sie die ENI-ID. Verwenden Sie den ARN der Aufgabe für den `tasks`-Parameter.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/service/EXAMPLE
```

Die Anhangsinformationen sind in der Ausgabe aufgeführt.

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                    ]
                }
…
}
```

Beschreiben Sie die ENI, um die öffentliche IP-Adresse abzurufen.

```
aws ec2 describe-network-interfaces --network-interface-id  eni-0fa40520aeEXAMPLE
```

Die öffentliche IP-Adresse befindet sich in der Ausgabe.

```
{
    "NetworkInterfaces": [
        {
            "Association": {
```

```
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-229-42-222.compute-1.amazonaws.com",
                "PublicIp": "198.51.100.2"
            },
…
}
```

Geben Sie die öffentliche IP-Adresse in Ihren Webbrowser ein. Sie sehen eine Webseite, die die Amazon-ECS-Beispielsanwendung anzeigt.

Die Testaufgabe wurde über ein privates Subnetz bereitgestellt

Beschreiben Sie die Aufgabe und suchen Sie nach `managedAgents`, um zu überprüfen, ob der `ExecuteCommandAgent` ausgeführt wird. Notieren Sie sich die `privateIPv4Address` für die spätere Verwendung.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/fargate-service/EXAMPLE
```

Die Informationen zum verwalteten Agenten werden in der Ausgabe aufgelistet.

```
{
    "tasks": [
       {
           "attachments": [
               {
                   "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                   "type": "ElasticNetworkInterface",
                   "status": "ATTACHED",
                   "details": [
                       {
                           "name": "subnetId",
                           "value": "subnetabcd1234"
                       },
                       {
                           "name": "networkInterfaceId",
                           "value": "eni-0fa40520aeEXAMPLE"
                       },
                       {
                           "name": "privateIPv4Address",
                           "value": "10.0.143.156"
                       }
```

```
                    ]
                }
            ],
    ...
    "containers": [
        {
        ...
       "managedAgents": [
                        {
                            "lastStartedAt": "2023-08-01T16:10:13.002000+00:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                ],
        ...
   }
```

Nachdem Sie sich vergewissert haben, dass `ExecuteCommandAgent` ausgeführt wird, können Sie den folgenden Befehl ausführen, um eine interaktive Shell für den Container in der Aufgabe auszuführen.

```
aws ecs execute-command --cluster fargate-cluster \
    --task  arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE  \
    --container  fargate-app \
    --interactive \
    --command "/bin/sh"
```

Nachdem die interaktive Shell ausgeführt wurde, führen Sie die folgenden Befehle aus, um cURL zu installieren.

```
apt update
```

```
apt install curl
```

Führen Sie nach der Installation von cURL den folgenden Befehl mit der privaten IP-Adresse aus, die Sie zuvor erhalten haben.

```
curl 10.0.143.156
```

Sie sollten das HTML-Äquivalent der Amazon-ECS-Beispielanwendungs-Webseite sehen.

```
<html>
    <head>
     <title>Amazon ECS Sample App</title>
     <style>body {margin-top: 40px; background-color: #333;} </style>
    </head>
      <body>
      <div style=color:white;text-align:center>
      <h1>Amazon ECS Sample App</h1>
      <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>
      </div>
      </body>
</html>
```

Schritt 8: Bereinigen

Wenn Sie mit diesem Tutorial fertig sind, sollten Sie die zugehörigen Ressourcen bereinigen, um zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen.

Löschen Sie den Service:

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Löschen Sie den Cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Erstellen einer Amazon ECS-Windows-Aufgabe für den Fargate-Starttyp mit dem AWS CLI

Mithilfe der folgenden Schritte können Sie über die AWS CLI einen Cluster einrichten, eine Aufgabendefinition registrieren, eine Windows-Aufgabe ausführen sowie andere allgemeine Szenarien in Amazon ECS umsetzen. Vergewissern Sie sich, dass Sie die neueste Version der AWS CLI-CLI verwenden. Weitere Informationen zur Aktualisierung auf die neueste Version finden Sie unter Installieren der AWS Command Line Interface.

Themen

- Voraussetzungen

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen zur Installation oder Aktualisierung von finden Sie AWS CLI unter Installation von AWS Command Line Interface.
- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben eine VPC und die zu verwendende Sicherheitsgruppe erstellt. In diesem Tutorial wird ein Container-Image verwendet, das auf Docker Hub gehostet wird, sodass Ihre Aufgabe über Internetzugang verfügen muss. Um Ihrer Aufgabe eine Route zum Internet zu geben, verwenden Sie eine der folgenden Optionen.
  - Verwenden Sie ein privates Subnetz mit einem NAT-Gateway, das über eine Elastic IP-Adresse verfügt.
  - Verwenden Sie ein öffentliches Subnetz und weisen Sie der Aufgabe eine öffentliche IP-Adresse zu.

  Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.

  Informationen zu Sicherheitsgruppen und -regeln finden Sie unter Standard-Sicherheitsgruppen für Ihre VPCs und Beispielregeln im Benutzerhandbuch zur Amazon Virtual Private Cloud.
- (Optional) AWS CloudShell ist ein Tool, das Kunden eine Befehlszeile bietet, ohne ihre eigene EC2-Instance erstellen zu müssen. Weitere Informationen finden Sie unter Was ist? AWS CloudShell im AWS CloudShell Benutzerhandbuch.

Schritt 1: Erstellen eines Clusters

Ihr Konto erhält standardmäßig ein `default` Cluster.

> **Note**
>
> Wenn Sie den bereitgestellten Cluster `default` verwenden, hat dies den Vorteil, dass Sie in den nachfolgenden Befehlen nicht die Option `--cluster` *`cluster_name`* angeben müssen. Wenn Sie Ihren eigenen, nicht standardmäßigen Cluster erstellen, müssen Sie `--cluster` *`cluster_name`* für jeden Befehl angeben, den Sie mit diesem Cluster verwenden möchten.

Erstellen Sie mit dem folgenden Befehl Ihren eigenen Cluster mit eindeutigem Namen:

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Ausgabe:

```
{
    "cluster": {
        "status": "ACTIVE",
        "statistics": [],
        "clusterName": "fargate-cluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Schritt 2: Anmelden einer Windows-Aufgabendefinition

Bevor Sie auf Ihrem Amazon-ECS-Cluster eine Windows-Aufgabe ausführen können, müssen Sie eine Aufgabendefinition anmelden. Aufgabendefinitionen sind Listen zusammengefasster Container. Das folgende Beispiel ist eine einfache Aufgabendefinition, die eine Webanwendung erstellt. Weitere Informationen zu den verfügbaren Parametern für die Aufgabendefinition finden Sie im Abschnitt Amazon-ECS-Aufgabendefinitionen.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

Das obige Beispiel-JSON kann auf zwei Arten AWS CLI an die übergeben werden: Sie können die Aufgabendefinition JSON als Datei speichern und sie mit der `--cli-input-json file://`*`path_to_file.json`* Option übergeben.

Verwendung einer JSON-Datei für Containerdefinitionen:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

Der Befehl register-task-definition gibt nach Abschluss der Registrierung eine Beschreibung der Aufgabendefinition zurück.

Schritt 3: Auflisten der Aufgabendefinitionen

Sie können die Aufgabendefinitionen für Ihr Konto jederzeit mit dem Befehl list-task-definitions auflisten. In der Ausgabe dieses Befehls werden die Werte `family` und `revision` angezeigt, die Sie beim Aufruf von run-task oder start-task gemeinsam verwenden können.

```
aws ecs list-task-definitions
```

Ausgabe:

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate-windows:1"
    ]
}
```

Schritt 4: Erstellen eines Services

Nachdem Sie eine Aufgabe für Ihr Konto registriert haben, können Sie einen Service für die registrierte Aufgabe in Ihrem Cluster erstellen. In diesem Beispiel erstellen Sie einen Service mit einer Instance der `sample-fargate:1`-Aufgabendefinition, die in Ihrem Cluster ausgeführt wird. Die Aufgabe erfordert eine Route zum Internet, daher gibt es zwei Möglichkeiten, dies zu erreichen. Eine Möglichkeit besteht darin, ein privates Subnetz zu verwenden, das mit einem NAT-Gateway mit einer Elastic IP-Adresse in einem öffentlichen Subnetz konfiguriert ist. Eine andere Möglichkeit besteht darin, ein öffentliches Subnetz zu verwenden und Ihrer Aufgabe eine öffentliche IP-Adresse zuzuweisen. Wir stellen beide Beispiele unten zur Verfügung.

Beispiel mit einem privaten Subnetz.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234]}"
```

Beispiel mit einem öffentlichen Subnetz.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

Der Befehl create-service gibt nach Abschluss der Registrierung eine Beschreibung der Aufgabendefinition zurück.

Schritt 5: Auflisten von Services

Listet die Services für Ihren Cluster auf. Der im vorherigen Abschnitt erstellte Service müsste angezeigt werden. Sie können den Servicenamen oder den vollständigen ARN, die/der von diesem Befehl zurückgegeben wird, später zur Beschreibung des Service verwenden.

```
aws ecs list-services --cluster fargate-cluster
```

Ausgabe:

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-service"
    ]
}
```

Schritt 6: Beschreibung des gerade ausgeführten Service

Beschreibt Sie den Service unter Verwendung des zuvor abgerufenen Servicenamen, um weitere Informationen über die Aufgabe zu erhalten.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

Wenn dies erfolgreich ist, wird eine Beschreibung der Servicefehler und Services zurückgegeben. Im Abschnitt „Services“ finden Sie beispielsweise Informationen zu Bereitstellungen, z. B. ob Aufgaben den Status ausgeführt oder ausstehend haben. Möglicherweise finden Sie auch Informationen zur Aufgabendefinition, zur Netzwerkkonfiguration und zu Ereignissen mit Zeitstempeln. Im Abschnitt „Fehler“ finden Sie Informationen zu Fehlern (falls vorhanden), die mit dem Aufruf verbunden

sind. Informationen zur Fehlerbehebung finden Sie unter Service-Ereignismeldungen. Weitere Informationen zur Servicebeschreibung finden Sie unter Beschreiben von Services.


```
{
    "services": [
        {
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate-windows:1",
            "pendingCount": 2,
            "launchType": "FARGATE",
            "loadBalancers": [],
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "placementConstraints": [],
            "createdAt": 1510811361.128,
            "desiredCount": 2,
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "DISABLED"
                }
            },
            "platformVersion": "LATEST",
            "serviceName": "fargate-service",
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
```

```
                        ],
                        "securityGroups": [
                            "sg-abcd1234"
                        ],
                        "assignPublicIp": "DISABLED"
                    }
                },
                "pendingCount": 2,
                "launchType": "FARGATE",
                "createdAt": 1510811361.128,
                "desiredCount": 2,
                "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate-windows:1",
                "updatedAt": 1510811361.128,
                "platformVersion": "0.0.1",
                "id": "ecs-svc/9223370526043414679",
                "runningCount": 0
            }
        ],
        "events": [
            {
                "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                "createdAt": 1510811841.408
            },
            {
                "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                "createdAt": 1510811601.938
            },
            {
                "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                "createdAt": 1510811364.691
            }
        ],
        "runningCount": 0,
        "placementStrategy": []
    }
],
"failures": []
```

```
}
```

Schritt 7: Bereinigen

Wenn Sie mit diesem Tutorial fertig sind, sollten Sie die zugehörigen Ressourcen bereinigen, um zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen.

Löschen Sie den Service:

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Löschen Sie den Cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Erstellen einer Amazon ECS-Aufgabe für den EC2-Starttyp mit dem AWS CLI

Mithilfe der folgenden Schritte können Sie über die AWS CLI einen Cluster einrichten, eine Aufgabendefinition registrieren, eine Aufgabe ausführen sowie andere allgemeine Szenarien in Amazon ECS umsetzen. Verwenden Sie die neueste Version von AWS CLI. Weitere Informationen zur Aktualisierung auf die neueste Version finden Sie unter Installieren der AWS Command Line Interface.

Themen

- Schritt 9: Beschreibung der gerade ausgeführten Aufgabe

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen zur Installation oder Aktualisierung von finden Sie AWS CLI unter Installation von AWS Command Line Interface.
- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Sie haben eine VPC und die zu verwendende Sicherheitsgruppe erstellt. Weitere Informationen finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.
- (Optional) AWS CloudShell ist ein Tool, das Kunden eine Befehlszeile bietet, ohne ihre eigene EC2-Instance erstellen zu müssen. Weitere Informationen finden Sie unter Was ist? AWS CloudShell im AWS CloudShell Benutzerhandbuch.

Schritt 1: Erstellen eines Clusters

Standardmäßig erhält Ihr Konto beim Start Ihrer ersten Container-Instance einen Cluster `default`.

> **Note**
>
> Wenn Sie den bereitgestellten Cluster `default` verwenden, hat dies den Vorteil, dass Sie in den nachfolgenden Befehlen nicht die Option `--cluster` *`cluster_name`* angeben müssen. Wenn Sie Ihren eigenen, nicht standardmäßigen Cluster erstellen, müssen Sie `--cluster` *`cluster_name`* für jeden Befehl angeben, den Sie mit diesem Cluster verwenden möchten.

Erstellen Sie mit dem folgenden Befehl Ihren eigenen Cluster mit eindeutigem Namen:

```
aws ecs create-cluster --cluster-name MyCluster
```

Ausgabe:

```
{
    "cluster": {
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/MyCluster"
    }
}
```

Schritt 2: Launchen einer Instance mit dem Amazon-ECS-AMI

In Ihrem Cluster muss eine Amazon-ECS-Container-Instance enthalten sein, damit Sie Aufgaben darin ausführen können. Wenn es in Ihrem Cluster keine Container-Instances gibt, beachten Sie die Informationen im Abschnitt Starten einer Amazon ECS Linux-Container-Instance.

Schritt 3: Auflisten von Container-Instances

Innerhalb von wenigen Minuten nach dem Launch Ihrer Container-Instance meldet der Amazon-ECS-Agent die Instance bei Ihrem Standard-Cluster an. Die Container-Instances in einem Cluster können mit dem folgenden Befehl aufgelistet werden:

```
aws ecs list-container-instances --cluster default
```

Ausgabe:

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:container-instance/container_instance_ID"
    ]
}
```

Schritt 4: Beschreiben Ihrer Container-Instance

Wenn Sie den ARN oder die ID einer Container-Instance haben, können Sie mit dem Befehl describe-container-instances wichtige Informationen zu der Instance abrufen, wie z. B. die verbleibenden und registrierten CPU- und Speicherressourcen.

```
aws ecs describe-container-instances --cluster default --container-
instances container_instance_ID
```

Ausgabe:


```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "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
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
```

```
            "ec2InstanceId": "instance_id",
            "agentConnected": true,
            "containerInstanceArn": "arn:aws:ecs:us-west-2:aws_account_id:container-
instance/container_instance_ID",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "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
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "runningTasksCount": 0,
            "attributes": [
```

```
                {
                    "name": "com.amazonaws.ecs.capability.privileged-container"
                },
                {
                    "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": "com.amazonaws.ecs.capability.logging-driver.json-file"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                }
            ],
            "versionInfo": {
                "agentVersion": "1.5.0",
                "agentHash": "b197edd",
                "dockerVersion": "DockerVersion: 1.7.1"
            }
        }
    ]
}
```

Hier finden Sie auch die Amazon-EC2-Instance-ID, mit der Sie die Instance in der Amazon-EC2-Konsole oder mit dem aws ec2 describe-instances --instance-id ***instance_id***-Befehl überwachen können.

Schritt 5: Registrieren einer Aufgabendefinition

Bevor Sie auf Ihrem ECS-Cluster eine Aufgabe ausführen können, müssen Sie eine Aufgabendefinition registrieren. Aufgabendefinitionen sind Listen zusammengefasster Container. Im folgenden Beispiel sehen Sie eine einfache Aufgabendefinition, die ein `busybox`-Image aus dem Docker-Hub verwendet und sich einfach 360 Sekunden im Ruhezustand befindet. Weitere Informationen zu den verfügbaren Parametern für die Aufgabendefinition finden Sie im Abschnitt Amazon-ECS-Aufgabendefinitionen.

```
{
```

```
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "busybox",
            "cpu": 10,
            "command": [
                "sleep",
                "360"
            ],
            "memory": 10,
            "essential": true
        }
    ],
    "family": "sleep360"
}
```

Das obige Beispiel-JSON kann auf zwei Arten AWS CLI an die übergeben werden: Sie können die Aufgabendefinition JSON als Datei speichern und sie mit der `--cli-input-json file://path_to_file.json` Option übergeben. Oder Sie können die Anführungszeichen in den JSON-Daten mit Escapezeichen versehen und die JSON-Containerdefinitionen wie im Beispiel unten dargestellt an die Befehlszeile übergeben. Wenn Sie die Containerdefinitionen in der Befehlszeile übergeben möchten, müssen Sie in dem Befehl zusätzlich einen Parameter `--family` angeben, damit mehrere Versionen Ihrer Aufgabendefinition miteinander verbunden bleiben.

Verwendung einer JSON-Datei für Containerdefinitionen:

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/sleep360.json
```

Verwendung einer JSON-Zeichenfolge für Containerdefinitionen:

```
aws ecs register-task-definition --family sleep360 --container-definitions "[{\"name
\":\"sleep\",\"image\":\"busybox\",\"cpu\":10,\"command\":[\"sleep\",\"360\"],\"memory
\":10,\"essential\":true}]"
```

Der Befehl register-task-definition gibt nach Abschluss der Registrierung eine Beschreibung der Aufgabendefinition zurück.

```
{
    "taskDefinition": {
        "volumes": [],
```

```
        "taskDefinitionArn": "arn:aws:ec2:us-east-1:aws_account_id:task-definition/
sleep360:1",
        "containerDefinitions": [
            {
                "environment": [],
                "name": "sleep",
                "mountPoints": [],
                "image": "busybox",
                "cpu": 10,
                "portMappings": [],
                "command": [
                    "sleep",
                    "360"
                ],
                "memory": 10,
                "essential": true,
                "volumesFrom": []
            }
        ],
        "family": "sleep360",
        "revision": 1
    }
}
```

Schritt 6: Auflisten der Aufgabendefinitionen

Sie können die Aufgabendefinitionen für Ihr Konto jederzeit mit dem Befehl list-task-definitions auflisten. In der Ausgabe dieses Befehls werden die Werte `family` und `revision` angezeigt, die Sie beim Aufruf von run-task oder start-task gemeinsam verwenden können.

```
aws ecs list-task-definitions
```

Ausgabe:

```
{
    "taskDefinitionArns": [
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep300:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep300:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:3",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:4",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:5",
```

```
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:6"
    ]
}
```

Schritt 7: Ausführen einer Task

Nachdem Sie eine Aufgabe für Ihr Konto registriert und eine für Ihren Cluster registrierte Container-Instance gestartet haben, können Sie die registrierte Aufgabe in Ihrem Cluster ausführen. In diesem Beispiel stellen Sie eine einzelne Instance der Aufgabendefinition `sleep360:1` in Ihren Standardcluster.

```
aws ecs run-task --cluster default --task-definition sleep360:1 --count 1
```

Ausgabe:

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "sleep"
                    }
                ]
            },
            "lastStatus": "PENDING",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/container_instance_ID",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/default",
            "desiredStatus": "RUNNING",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
sleep360:1",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-
east-1:aws_account_id:container/container_ID",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
                    "lastStatus": "PENDING",
                    "name": "sleep"
                }
            ]
```

```
        }
    ]
}
```

Schritt 8: Auflisten der Aufgaben

Listen Sie die Aufgaben für Ihren Cluster auf. Die im vorherigen Abschnitt ausgeführte Aufgabe müsste angezeigt werden. Sie können die Aufgaben-ID oder den vollständigen ARN, die/der von diesem Befehl zurückgegeben wird, später zur Beschreibung der Aufgabe verwenden.

```
aws ecs list-tasks --cluster default
```

Ausgabe:

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Schritt 9: Beschreibung der gerade ausgeführten Aufgabe

Beschreiben Sie die Aufgabe und verwenden Sie dazu die zuvor abgerufene Aufgaben-ID, um weitere Informationen über die Aufgabe zu erhalten.

```
aws ecs describe-tasks --cluster default --task task_ID
```

Ausgabe:

```
{
    "failures": [],
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "sleep"
                    }
                ]
            },
```

```
            "lastStatus": "RUNNING",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/container_instance_ID",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/default",
            "desiredStatus": "RUNNING",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
sleep360:1",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-
east-1:aws_account_id:container/container_ID",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
                    "lastStatus": "RUNNING",
                    "name": "sleep",
                    "networkBindings": []
                }
            ]
        }
    ]
}
```

Konfiguration von Amazon ECS für die Überwachung von CloudWatch Ereignissen

Erfahren Sie, wie Sie eine einfache Lambda-Funktion einrichten, die auf Aufgabenereignisse wartet und diese in einen CloudWatch Log-Log-Stream schreibt.

Voraussetzung: Einrichten eines Testclusters

Wenn Sie über kein aktives Cluster verfügen, mit dem Sie Ereignisse erfassen, führen Sie die Schritte unter the section called “Einen Cluster für den Fargate-Starttyp erstellen” aus, um eines zu erstellen. Am Ende dieses Tutorials führen Sie eine Aufgabe auf diesem Cluster aus, um zu testen, ob Sie Ihre Lambda-Funktion korrekt konfiguriert haben.

Schritt 1: Erstellen der Lambda-Funktion

In diesem Verfahren erstellen Sie eine einfache Lambda-Funktion, die als Ziel für Amazon-ECS-Ereignis-Stream-Nachrichten dient.

1. Öffnen Sie die AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.

2. Wählen Sie Funktion erstellen.
3. Verfahren Sie auf dem Bildschirm Author from scratch wie folgt:
   a. Geben Sie für Name einen Wert ein.
   b. Wählen Sie für Runtime (Laufzeit) Ihre Python-Version aus, etwa Python 3.9.
   c. Wählen Sie unter Role (Rolle) die Option Create a new role with basic Lambda permissions (Eine neue Rolle mit den grundlegenden Lambda-Berechtigungen erstellen) aus.
4. Wählen Sie Funktion erstellen.
5. Bearbeiten Sie im Bereich Function code den Beispiel-Code entsprechend dem folgenden Beispiel:

```
import json

def lambda_handler(event, context):
    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source
 type of: aws.ecs")

    print('Here is the event:')
    print(json.dumps(event))
```

   Dies ist eine einfache Python 3.9-Funktion, mit der das von Amazon ECS gesendete Ereignis gedruckt wird. Wenn alles korrekt konfiguriert ist, sehen Sie am Ende dieses Tutorials, dass die Ereignisdetails im CloudWatch Log-Protokollstream angezeigt werden, der dieser Lambda-Funktion zugeordnet ist.
6. Wählen Sie Speichern.

Schritt 2: Registrieren von Ereignisregeln

Als Nächstes erstellen Sie eine CloudWatch Ereignisregel, die Aufgabenereignisse erfasst, die aus Ihren Amazon ECS-Clustern stammen. Diese Regel erfasst alle Ereignisse, die aus allen Clustern innerhalb des Kontos stammen, in dem sie definiert wurde. Die Aufgabennachrichten selbst enthalten Informationen über die Ereignisquelle, einschließlich dem Cluster, in dem sie sich befindet. Sie können diese verwenden, um Ereignisse programmgesteuert zu filtern und zu sortieren.

> **Note**
>
> Wenn Sie die verwenden, AWS Management Console um eine Ereignisregel zu erstellen, fügt die Konsole automatisch die IAM-Berechtigungen hinzu, die erforderlich sind, um CloudWatch Events die Berechtigung zum Aufrufen Ihrer Lambda-Funktion zu erteilen. Wenn Sie mithilfe von eine Ereignisregel erstellen AWS CLI, müssen Sie diese Berechtigung explizit erteilen. Weitere Informationen finden Sie unter Ereignisse und Ereignismuster im Amazon CloudWatch Events-Benutzerhandbuch.

Umleiten von Ereignissen an Ihre Lambda-Funktion

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Events (Ereignisse), Rules (Regeln), Create rule (Regel erstellen) aus.
3. Wählen Sie für Event Source als Ereignisquelle ECS aus. Standardmäßig gilt die Regel für alle Amazon-ECS-Ereignisse all Ihrer Amazon-ECS-Gruppen. Alternativ können Sie bestimmte Ereignisse oder eine bestimmte Amazon-ECS-Gruppe auswählen.
4. Wählen Sie für Targets (Ziele) die Option Add target (Ziel hinzufügen) und für Target type (Zieltyp) die Option Lambda function (Lambda-Funktion) und danach Ihre Lambda-Funktion aus.
5. Wählen Sie Details konfigurieren.
6. Geben Sie für Rule definition einen Namen und eine Beschreibung für Ihre Regel ein und wählen Sie Create rule aus.

Schritt 3: Erstellen einer Aufgabendefinition

Erstellen Sie eine Aufgabendefinition.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task Definitions aus.
3. Wählen Sie Create new Task Definition (Neue Aufgabendefinition erstellen), Create new revision with JSON (Neue Revision mit JSON) erstellen.
4. Kopieren Sie die folgende Beispielaufgabendefinition, fügen Sie sie in das Feld ein, und wählen Sie dann Save (Speichern).

```
{
```

```
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ],
            "cpu": 10,
            "memory": 300,
            "image": "httpd:2.4",
            "name": "simple-app"
        }
    ],
    "family": "console-sample-app-static"
}
```

5. Wählen Sie Erstellen.

Schritt 4: Testen Ihrer Regel

Schließlich erstellen Sie eine CloudWatch Ereignisregel, die Aufgabenereignisse erfasst, die aus Ihren Amazon ECS-Clustern stammen. Diese Regel erfasst alle Ereignisse, die aus allen Clustern innerhalb des Kontos stammen, in dem sie definiert wurde. Die Aufgabennachrichten selbst enthalten Informationen über die Ereignisquelle, einschließlich dem Cluster, in dem sie sich befindet. Sie können diese verwenden, um Ereignisse programmgesteuert zu filtern und zu sortieren.

Testen Ihrer Regel

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie Task definitions (Aufgabendefinitionen).
3. Wählen Sie console-sample-app-static und wählen Sie dann Deploy (Bereitstellen), Run new task (Neue Aufgabe ausführen) aus.
4. Wählen Sie für Cluster die Option Standard und wählen Sie dann Deploy (Bereitstellen) aus.
5. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
6. Wählen Sie im Navigationsbereich Logs und die Protokollgruppe für Ihre Lambda-Funktion aus (z. B. /aws/lambda/*my-function*).
7. Wählen Sie einen Protokollstream aus, um die Ereignisdaten anzuzeigen.

Senden von Amazon Simple Notification Service-Benachrichtigungen für Ereignisse, bei denen Amazon ECS-Aufgaben gestoppt wurden

Konfigurieren Sie eine EventBridge Amazon-Ereignisregel, die nur Aufgabenereignisse erfasst, bei denen die Ausführung der Aufgabe beendet wurde, weil einer ihrer wichtigsten Container beendet wurde. Das Ereignis sendet nur Aufgabenereignisse mit einer bestimmten `stoppedReason`-Eigenschaft an das zugewiesene Amazon SNS-Thema.

Voraussetzung: Einrichten eines Testclusters

Wenn Sie keinen laufenden Cluster haben, von dem Sie Ereignisse erfassen können, folgen Sie den Schritten unter Erste Schritte mit der Konsole unter Verwendung von Linux-Containern auf AWS Fargate, um einen zu erstellen. Am Ende dieses Tutorials führen Sie eine Aufgabe auf diesem Cluster aus, um zu testen, ob Sie Ihr Amazon SNS-Thema und Ihre Amazon EventBridge SNS-Regel korrekt konfiguriert haben.

Voraussetzung: Berechtigungen für Amazon SNS konfigurieren

Verwenden Sie EventBridge die Befehle aws sns get-topic-attributes und aws sns, um die Veröffentlichung in einem Amazon SNS-Thema zu ermöglichen. set-topic-attributes

Informationen zum Hinzufügen der Berechtigung finden Sie unter Amazon-SNS-Berechtigungen im Entwicklerhandbuch für Amazon Simple Notification Service

Fügen Sie die folgenden Berechtigungen hinzu:

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sns: Publish",
  "Resource": "arn:aws:sns:region:account-id:TaskStoppedAlert",
}
```

Schritt 1: Erstellen und Abonnieren eines Amazon-SNS-Themas

Mit diesem Tutorial konfigurieren Sie ein Amazon SNS-Thema, das als Ereignisziel für Ihre neue Ereignisregel dient.

Informationen zum Erstellen und Abonnieren eines Amazon-SNS-Themas finden Sie unter Erste Schritte mit Amazon SNS im Entwicklerhandbuch zu Amazon Simple Notification Service und verwenden Sie die folgende Tabelle, um zu bestimmen, welche Optionen auszuwählen sind.

Option	Value
Typ	Standard
Name	TaskStoppedWarnung
Protokoll	Email
Endpunkt	Eine E-Mail-Adresse, auf die Sie aktuell Zugriff haben

Schritt 2: Registrieren von Ereignisregeln

Als nächstes registrieren Sie eine Ereignisregel, die nur „Aufgabe angehalten“-Ereignisse für Aufgaben mit angehaltenen Containern erfasst.

Informationen zum Erstellen und Abonnieren eines Amazon SNS SNS-Themas finden Sie unter Erstellen einer Regel in Amazon EventBridge im EventBridge Amazon-Benutzerhandbuch. Verwenden Sie die folgende Tabelle, um zu bestimmen, welche Optionen Sie auswählen müssen.

Option	Wert
Regeltyp	Regel mit einem Ereignismuster
Ereignisquelle	AWS Veranstaltungen oder EventBridge Partnerveranstaltungen
Ereignismuster	Benutzerdefiniertes Muster (JSON-Editor)

Ereignismuster	<pre>{
	"source":[
	"aws.ecs"
	],
	"detail-type":[
	"ECS Task State Change"
	],
	"detail":{
	"lastStatus":[
	"STOPPED"
	],
	"stoppedReason":[
	"Essential container in task exited"
	]
	}
	}</pre>

Zieltyp	AWS Service
Ziel	SNS-Thema
Thema	TaskStoppedAlert (Das Thema, das Sie in Schritt 1 erstellt haben)

Schritt 3: Testen Ihrer Regel

Stellen Sie sicher, dass die Regel funktioniert, indem Sie eine Aufgabe ausführen, die kurz nach dem Start beendet wird. Wenn Ihre Ereignisregel korrekt konfiguriert ist, erhalten Sie innerhalb weniger Minuten eine E-Mail-Nachricht mit dem Ereignistext. Wenn Sie über eine vorhandene Aufgabendefinition verfügen, die die Regelanforderungen erfüllen kann, führen Sie eine Aufgabe mit dieser aus. Wenn dies nicht der Fall ist, führen Sie die folgenden Schritte durch die Registrierung einer Fargate-Aufgabendefinition und deren Ausführung.

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie im Navigationsbereich Task definitions (Aufgabendefinitionen) aus.
3. Wählen Sie Create new task definition (Neue Aufgabendefinition erstellen), Create new task definition with JSON (Neue Aufgabendefinition mit JSON) erstellen.
4. Bearbeiten Sie im JSON-Editorfeld Ihre JSON-Datei und kopieren Sie Folgendes in den Editor.

```
{
   "containerDefinitions":[
      {
         "command":[
            "sh",
            "-c",
            "sleep 5"
         ],
         "essential":true,
         "image":"amazonlinux:2",
         "name":"test-sleep"
      }
   ],
   "cpu":"256",
   "executionRoleArn":"arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family":"fargate-task-definition",
   "memory":"512",
   "networkMode":"awsvpc",
   "requiresCompatibilities":[
      "FARGATE"
   ]
}
```

5. Wählen Sie Erstellen.

So führen Sie eine Aufgabe über die Konsole aus

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
2. Wählen Sie auf der Seite Cluster den Cluster aus, den Sie in den Voraussetzungen erstellt haben.
3. Von der Registerkarte Tasks (Aufgaben) wählen Sie Ausführen einer neuen Aufgabe.
4. Für Anwendungstyp, wählen Sie Aufgabe aus.
5. Wählen Sie für Aufgabendefinition fargate-task-definition aus.
6. Geben Sie für Desired tasks (Gewünschte Aufgaben) die Anzahl der Aufgaben an, die gestartet werden sollen.
7. Wählen Sie Erstellen.

Verkettung mehrzeiliger Amazon ECS-Protokollnachrichten oder Stack-Trace-Protokollnachrichten

Ab Version 2.22.0 von Fluent Bit ist ein Mehrzeilenfilter enthalten. AWS Der Mehrzeilenfilter hilft bei der Verkettung von Protokollmeldungen, die ursprünglich zu einem Kontext gehören, aber auf mehrere Datensätze oder Protokollzeilen aufgeteilt wurden. Weitere Informationen zum mehrzeiligen Filter finden Sie in der Fluent-Bit-Dokumentation.

Häufige Beispiele für geteilte Protokollmeldungen sind:

- Stack-Traces.
- Anwendungen, die Protokolle auf mehreren Zeilen drucken.
- Protokollmeldungen, die geteilt wurden, weil sie länger waren als die angegebene maximale Puffergröße der Laufzeit. Sie können Protokollnachrichten, die nach der Container-Laufzeit aufgeteilt sind, verketten, indem Sie dem Beispiel unter folgen: Beispiel GitHub: FireLens Concatenate Partial/Split Container Logs.

Erforderliche IAM-Berechtigungen

Sie verfügen über die erforderlichen IAM-Berechtigungen, damit der Container-Agent die Container-Images aus Amazon ECR abrufen kann und der Container Protokolle an CloudWatch Logs weiterleitet.

Für diese Berechtigungen müssen Sie über die folgenden Rollen verfügen:

- Eine Aufgaben-IAM-Rolle.
- Eine IAM-Rolle zur Aufgabenausführung.

So verwenden Sie den JSON-Richtlinieneditor zum Erstellen einer Richtlinie

1. Melden Sie sich bei der IAM-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "logs:CreateLogStream",
   "logs:CreateLogGroup",
   "logs:PutLogEvents"
  ],
  "Resource": "*"
 }]
}
```

6. Wählen Sie Weiter aus.

> **Note**
>
> Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter Richtlinienrestrukturierung im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtlinienname einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Festlegen, wann die Einstellung für mehrzeiliges Protokoll verwendet werden soll

Im Folgenden finden Sie Beispiele für Protokollausschnitte, die Sie in der CloudWatch Logs-Konsole mit der Standardprotokolleinstellung sehen. Sie können sich die Zeile ansehen, die mit `log` beginnt, um festzustellen, ob Sie den Mehrzeilenfilter benötigen. Wenn der Kontext derselbe ist, können Sie die Einstellung für das mehrzeilige Protokoll verwenden. In diesem Beispiel ist der Kontext „com.myproject.model“. MyProject“.

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-
app", "source=": "stdout", "log": ": "     at com.myproject.modele.
(MyProject.badMethod.java:22)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at com.myproject.model.MyProject.badMethod(MyProject.java:22)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app", "stdout",
 "log": ": "     at com.myproject.modele.(MyProject.oneMoreMethod.java:18)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
```

```
      "log": ": "     at
 com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE,
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

Nachdem Sie die mehrzeilige Protokolleinstellung verwendet haben, sieht die Ausgabe ähnlich aus wie im Beispiel unten.

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app",
 "stdout",...
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log:    "September 20, 2022 06:41:48 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
    at com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)
 com.myproject.module.MyProject.main(MyProject.java:6)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:2"
     }
```

Analyse und Verkettung von Optionen

Um Protokolle zu analysieren und Zeilen zu verketten, die aufgrund von Zeilenumbrüchen geteilt wurden, können Sie eine dieser beiden Optionen verwenden.

- Verwenden Sie Ihre eigene Parserdatei, die die Regeln zum Analysieren und Verketten von Zeilen enthält, die zu derselben Nachricht gehören.
- Einen integrierten Parser für Fluent Bit verwenden. Eine Liste der Sprachen, die von den integrierten Parsern Fluent Bit unterstützt werden, finden Sie unter Fluent-Bit-Dokumentation.

Das folgende Tutorial führt Sie durch die Schritte für jeden Anwendungsfall. Die Schritte zeigen Ihnen, wie Sie mehrere Zeilen verketten und die Protokolle an Amazon senden. CloudWatch Sie können ein anderes Ziel für Ihre Protokolle angeben.

Beispiel: Verwenden eines Parsers, den Sie erstellen

In diesem Beispiel führen Sie die folgenden Schritte aus:

1. Erstellen und laden Sie das Image für einen Fluent-Bit-Container hoch.
2. Erstellen und laden Sie das Image für eine mehrzeilige Demo-Anwendung hoch, die einen mehrzeiligen Stack-Trace ausführt, fehlschlägt und generiert.
3. Erstellen Sie die Aufgabendefinition und führen Sie die Aufgabe aus.
4. Zeigen Sie die Protokolle an, um zu überprüfen, ob Nachrichten, die sich über mehrere Zeilen erstrecken, verkettet erscheinen.

Erstellen und Hochladen des Images für einen Fluent-Bit-Container

Dieses Image enthält die Parserdatei, in der Sie den regulären Ausdruck angeben, und eine Konfigurationsdatei, die auf die Parser-Datei verweist.

1. Erstellen Sie einen Ordner mit dem Namen `FluentBitDockerImage`.
2. Erstellen Sie innerhalb des Ordners eine Parserdatei, die die Regeln zum Analysieren des Protokolls und zum Verketten von Zeilen enthält, die zu derselben Nachricht gehören.

   a. Fügen Sie den folgenden Inhalt in die Parser-Datei ein:

```
[MULTILINE_PARSER]
    name          multiline-regex-test
    type          regex
    flush_timeout 1000
    #
    # Regex rules for multiline parsing
    # ---------------------------------
    #
    # configuration hints:
    #
    #  - first state always has the name: start_state
    #  - every field in the rule must be inside double quotes
    #
    # rules |   state name  | regex pattern                  | next state
```

```
    # ------|---------------|-------------------------------------------
    rule      "start_state"   "/(Dec \d+ \d+\:\d+\:\d+)(.*)/"  "cont"
    rule      "cont"          "/^\s+at.*/"                     "cont"
```

Wenn Sie Ihr Regex-Muster anpassen, empfehlen wir Ihnen, den Ausdruck mit einem Editor für reguläre Ausdrücke zu testen.

b. Speichern Sie die Datei als `parsers_multiline.conf`.

3. Erstellen Sie im `FluentBitDockerImage`-Ordner eine benutzerdefinierte Konfigurationsdatei, die auf die Parserdatei verweist, die Sie im vorherigen Schritt erstellt haben.

Weitere Informationen zur benutzerdefinierten Konfigurationsdatei finden Sie unter Angeben einer benutzerdefinierten Konfigurationsdatei im Entwicklerhandbuch für Amazon Elastic Container Service

a. Fügen Sie den folgenden Inhalt in die Datei ein:

```
[SERVICE]
    flush                 1
    log_level             info
    parsers_file          /parsers_multiline.conf

[FILTER]
    name                  multiline
    match                 *
    multiline.key_content log
    multiline.parser      multiline-regex-test
```

> **Note**
>
> Sie müssen den absoluten Pfad des Parsers verwenden.

b. Speichern Sie die Datei als `extra.conf`.

4. Erstellen Sie im `FluentBitDockerImage`-Ordner die Dockerfile mit dem Fluent-Bit-Image und den von Ihnen erstellten Parser- und Konfigurationsdateien.

a. Fügen Sie den folgenden Inhalt in die Datei ein:

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
```

```
ADD parsers_multiline.conf /parsers_multiline.conf
ADD extra.conf /extra.conf
```

b. Speichern Sie die Datei als `Dockerfile`.

5. Erstellen Sie mit der Dockerfile ein benutzerdefiniertes Fluent-Bit-Image mit dem Parser und benutzerdefinierten Konfigurationsdateien.

> **Note**
>
> Sie können die Parser-Datei und die Konfigurationsdatei an einer beliebigen Stelle im Docker-Image platzieren, es sei `/fluent-bit/etc/fluent-bit.conf` denn, dieser Dateipfad wird von verwendet. FireLens

a. Entwickeln Sie das Image: `docker build -t fluent-bit-multiline-image .`

Wobei: `fluent-bit-multiline-image` der Name für das Image in diesem Beispiel ist.

b. Überprüfen Sie, ob das Image korrekt erstellt wurde: `docker images –filter reference=fluent-bit-multiline-image`

Bei Erfolg zeigt die Ausgabe das Image und das `latest`-Tag.

6. Laden Sie das benutzerdefinierte Fluent-Bit-Image in Amazon Elastic Container Registry hoch.

a. Erstellen Sie ein Amazon-ECR-Repository zum Speichern des Images: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

Wobei: `fluent-bit-multiline-repo` der Name für das Repository und `us-east-1` die Region in diesem Beispiel ist.

Die Ausgabe gibt Ihnen die Details des neuen Repositorys.

b. Markieren Sie Ihr Image mit dem `repositoryUri`-Wert aus der vorherigen Ausgabe: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

Beispiel: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Führen Sie das Docker-Image aus, um zu überprüfen, ob es korrekt ausgeführt wurde: `docker images –filter reference=`*`repositoryUri`*

   In der Ausgabe ändert sich der Repository-Name von fluent-bit-multiline-repo zu. `repositoryUri`

d. Authentifizieren Sie sich bei Amazon ECR, indem Sie den Befehl `aws ecr get-login-password` ausführen und die Registry-ID angeben, bei der Sie sich authentifizieren möchten: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

   Beispiel: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

   Eine erfolgreiche Anmeldemeldung wird angezeigt.

e. Verschieben Sie das Image zu Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

   Beispiel: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Erstellen und Hochladen des Images für eine mehrzeilige Demo-Anwendung

Dieses Image enthält eine Python-Skriptdatei, die die Anwendung ausführt, und eine Beispielprotokolldatei.

Wenn Sie die Aufgabe ausführen, simuliert die Anwendung Ausführungen, schlägt dann fehl und erstellt einen Stack-Trace.

1. Erstellen Sie einen Ordner mit dem Namen `multiline-app`: `mkdir multiline-app`
2. Erstellen Sie eine Python-Skriptdatei.

   a. Erstellen Sie im `multiline-app`-Ordner eine Datei und nennen Sie sie `main.py`.

   b. Fügen Sie den folgenden Inhalt in die Datei ein:

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()
```

```
count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Speichern Sie die `main.py`-Datei.

3. Erstellen Sie eine Beispielprotokolldatei.

   a. Erstellen Sie im `multiline-app`-Ordner eine Datei und nennen Sie sie `test.log`.

   b. Fügen Sie den folgenden Inhalt in die Datei ein:

```
single line...
Dec 14 06:41:08 Exception in thread "main" java.lang.RuntimeException:
 Something has gone wrong, aborting!
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)
    at com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)
    at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)
    at com.myproject.module.MyProject.someMethod(MyProject.java:10)
    at com.myproject.module.MyProject.main(MyProject.java:6)
another line...
```

   c. Speichern Sie die `test.log`-Datei.

4. Erstellen Sie innerhalb des Ordners `multiline-app` die Docker-Datei.

   a. Fügen Sie den folgenden Inhalt in die Datei ein:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin
```

```
COPY main.py .

CMD ["python3", "main.py"]
```

b. Speichern Sie die `Dockerfile`-Datei.

5. Erstellen Sie mit der Dockerfile ein Image.

   a. Entwickeln Sie das Image: `docker build -t multiline-app-image .`

      Wobei: `multiline-app-image` der Name für das Image in diesem Beispiel ist.

   b. Überprüfen Sie, ob das Image korrekt erstellt wurde: `docker images –filter reference=multiline-app-image`

      Bei Erfolg zeigt die Ausgabe das Image und das `latest`-Tag.

6. Laden Sie das Image in Amazon-Elastic-Container-Registry hoch.

   a. Erstellen Sie ein Amazon-ECR-Repository zum Speichern des Images: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Wobei: `multiline-app-repo` der Name für das Repository und `us-east-1` die Region in diesem Beispiel ist.

      Die Ausgabe gibt Ihnen die Details des neuen Repositorys. Notieren Sie sich den `repositoryUri`-Wert, da Sie ihn in den nächsten Schritten benötigen.

   b. Markieren Sie Ihr Image mit dem `repositoryUri`-Wert aus der vorherigen Ausgabe: `docker tag` *`multiline-app-image repositoryUri`*

      Beispiel: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

   c. Führen Sie das Docker-Image aus, um zu überprüfen, ob es korrekt ausgeführt wurde: `docker images –filter reference=`*`repositoryUri`*

      In der Ausgabe ändert sich der Repository-Name von `multiline-app-repo` in den `repositoryUri`-Wert.

   d. Verschieben Sie das Image zu Amazon ECR: `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

      Beispiel: `docker push` *`xxxxxxxxxxxx`*`.dkr.ecr.`*`us-east-1`*`.amazonaws.com/multiline-app-repo`

Erstellen einer Aufgabendefinition und Ausführen der Aufgabe

1. Erstellen Sie eine Aufgabendefinitionsdatei mit dem Dateinamen `multiline-task-definition.json`.
2. Fügen Sie den folgenden Inhalt in die `multiline-task-definition.json`-Datei ein:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
```

```
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Ersetzen Sie Folgendes in der `multiline-task-definition.json`-Aufgabendefinition:

a. *`task role ARN`*

Um den Aufgabenrollen-ARN zu finden, gehen Sie zur IAM-Konsole. Wählen Sie Roles (Rollen) und suchen Sie die `ecs-task-role-for-firelens`-Aufgabenrolle, die Sie erstellt haben. Wählen Sie die Rolle aus und kopieren Sie den ARN, der im Abschnitt Summary (Zusammenfassung) angezeigt wird.

b. *`execution role ARN`*

Verwenden Sie die IAM-Konsole, um die Ausführungsrollen-ARN zu finden. Wählen Sie Roles (Rollen) aus und finden Sie die `ecsTaskExecutionRole`-Rolle. Wählen Sie die Rolle aus und kopieren Sie den ARN, der im Abschnitt Summary (Zusammenfassung) angezeigt wird.

c. *`aws_account_id`*

Um Ihr `aws_account_id` zu finden, melden Sie sich bei AWS Management Console an. Wählen Sie oben rechts Ihren Benutzernamen und kopieren Sie Ihre Konto-ID.

d. *`us-east-1`*

Ersetzen Sie die Region bei Bedarf.

3. Registrieren Sie die Aufgabendefinitionsdatei: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region` *`region`*
4. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
5. Wählen Sie im Navigationsbereich Task Definitions (Aufgabendefinitionen) und dann die `firelens-example-multiline`-Familie aus, da wir die Aufgabendefinition für diese Familie in der ersten Zeile der Aufgabendefinition oben registriert haben.
6. Wählen Sie die neueste Version aus.
7. Wählen Sie Bereitstellen, Aufgabe ausführen aus.
8. Wählen Sie auf der Seite Task ausführen für Cluster den Cluster aus, und wählen Sie dann unter Netzwerk für Subnetze die verfügbaren Subnetze für Ihre Aufgabe aus.

9. Wählen Sie Erstellen.

Stellen Sie sicher, dass mehrzeilige Protokollnachrichten in Amazon verkettet CloudWatch angezeigt werden

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/cloudwatch/ CloudWatch .
2. Erweitern Sie im Navigationsbereich Logs (Protokolle) und wählen Sie Log groups (Protokollgruppen) aus.
3. Wählen Sie die Protokollgruppe `multiline-test/applicatio` aus.
4. Wählen Sie das Protokoll aus. Nachrichten anzeigen. Zeilen, die mit den Regeln in der Parser-Datei übereinstimmen, werden verkettet und werden als einzelne Nachricht angezeigt.

   Das folgende Protokoll-Snippet zeigt Zeilen an, die in einem einzigen Java-Stack-Trace-Ereignis verkettet sind:

```
{
    "container_id": "xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
 at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
 com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)\n
 at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)\n
 at com.myproject.module.MyProject.someMethod(MyProject.java:10)\n    at
 com.myproject.module.MyProject.main(MyProject.java:6)",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

   Das folgende Protokoll-Snippet zeigt, wie dieselbe Meldung mit nur einer einzigen Zeile angezeigt wird, wenn Sie einen Amazon-ECS-Container ausführen, der nicht für die Verkettung von mehrzeiligen Protokollmeldungen konfiguriert ist.

```
{
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!",
    "container_id": "xxxxxx-xxxxxx",
```

```
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
}
```

Beispiel: Verwenden eines integrierten Parsers für Fluent Bit

In diesem Beispiel führen Sie die folgenden Schritte aus:

1. Erstellen und laden Sie das Image für einen Fluent-Bit-Container hoch.
2. Erstellen und laden Sie das Image für eine mehrzeilige Demo-Anwendung hoch, die einen mehrzeiligen Stack-Trace ausführt, fehlschlägt und generiert.
3. Erstellen Sie die Aufgabendefinition und führen Sie die Aufgabe aus.
4. Zeigen Sie die Protokolle an, um zu überprüfen, ob Nachrichten, die sich über mehrere Zeilen erstrecken, verkettet erscheinen.

Erstellen und Hochladen des Images für einen Fluent-Bit-Container

Dieses Image enthält eine Konfigurationsdatei, die auf den Fluent-Bit-Parser verweist.

1. Erstellen Sie einen Ordner mit dem Namen `FluentBitDockerImage`.
2. Erstellen Sie innerhalb des `FluentBitDockerImage`-Ordners eine benutzerdefinierte Konfigurationsdatei, die auf die integrierte Parser-Datei von Fluent Bit verweist.

   Weitere Informationen zur benutzerdefinierten Konfigurationsdatei finden Sie unter Angeben einer benutzerdefinierten Konfigurationsdatei im Entwicklerhandbuch für Amazon Elastic Container Service

   a. Fügen Sie den folgenden Inhalt in die Datei ein:

   ```
   [FILTER]
       name                  multiline
       match                 *
       multiline.key_content log
       multiline.parser      go
   ```

   b. Speichern Sie die Datei als `extra.conf`.

3. Erstellen Sie im `FluentBitDockerImage`-Ordner die Dockerfile mit dem Fluent-Bit-Image und den von Ihnen erstellten Parser- und Konfigurationsdateien.

   a. Fügen Sie den folgenden Inhalt in die Datei ein:

   ```
   FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
   ADD extra.conf /extra.conf
   ```

   b. Speichern Sie die Datei als `Dockerfile`.

4. Erstellen Sie mit der Dockerfile ein benutzerdefiniertes Fluent-Bit-Image mit der enthaltenen benutzerdefinierten Konfigurationsdatei.

   > **Note**
   >
   > Sie können die Konfigurationsdatei an einer beliebigen Stelle im Docker-Image platzieren, es sei `/fluent-bit/etc/fluent-bit.conf` denn, dieser Dateipfad wird von FireLens verwendet.

   a. Entwickeln Sie das Image: `docker build -t fluent-bit-multiline-image .`

      Wobei: `fluent-bit-multiline-image` der Name für das Image in diesem Beispiel ist.

   b. Überprüfen Sie, ob das Image korrekt erstellt wurde: `docker images –filter reference=fluent-bit-multiline-image`

      Bei Erfolg zeigt die Ausgabe das Image und das `latest`-Tag.

5. Laden Sie das benutzerdefinierte Fluent-Bit-Image in Amazon Elastic Container Registry hoch.

   a. Erstellen Sie ein Amazon-ECR-Repository zum Speichern des Images: `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

      Wobei: `fluent-bit-multiline-repo` der Name für das Repository und `us-east-1` die Region in diesem Beispiel ist.

      Die Ausgabe gibt Ihnen die Details des neuen Repositorys.

   b. Markieren Sie Ihr Image mit dem `repositoryUri`-Wert aus der vorherigen Ausgabe: `docker tag` *`fluent-bit-multiline-image repositoryUri`*

Beispiel: `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Führen Sie das Docker-Image aus, um zu überprüfen, ob es korrekt ausgeführt wurde: `docker images –filter reference=`*`repositoryUri`*

In der Ausgabe ändert sich der Repository-Name von fluent-bit-multiline-repo zu. `repositoryUri`

d. Authentifizieren Sie sich bei Amazon ECR, indem Sie den Befehl `aws ecr get-login-password` ausführen und die Registry-ID angeben, bei der Sie sich authentifizieren möchten: `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

Beispiel: `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

Eine erfolgreiche Anmeldemeldung wird angezeigt.

e. Verschieben Sie das Image zu Amazon ECR: `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

Beispiel: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Erstellen und Hochladen des Images für eine mehrzeilige Demo-Anwendung

Dieses Image enthält eine Python-Skriptdatei, die die Anwendung ausführt, und eine Beispielprotokolldatei.

1. Erstellen Sie einen Ordner mit dem Namen `multiline-app`: `mkdir multiline-app`
2. Erstellen Sie eine Python-Skriptdatei.

   a. Erstellen Sie im `multiline-app`-Ordner eine Datei und nennen Sie sie `main.py`.

   b. Fügen Sie den folgenden Inhalt in die Datei ein:

```
import os
import time
file1 = open('/test.log', 'r')
```

```
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Speichern Sie die `main.py`-Datei.

3. Erstellen Sie eine Beispielprotokolldatei.

a. Erstellen Sie im `multiline-app`-Ordner eine Datei und nennen Sie sie `test.log`.

b. Fügen Sie den folgenden Inhalt in die Datei ein:

```
panic: my panic

goroutine 4 [running]:
panic(0x45cb40, 0x47ad70)
  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8 sp=0xc42003f710
 pc=0x422f7c
main.main.func1(0xc420024120)
  foo.go:6 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0
 sp=0xc42003f7d8 pc=0x44b4d1
created by main.main
  foo.go:5 +0x58

goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
```

```
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8
 sp=0xc420053fe0 pc=0x44b4d1

goroutine 2 [force gc (idle)]:
runtime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c
runtime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e
runtime.forcegchelper()
  /usr/local/go/src/runtime/proc.go:238 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8
 pc=0x424e5c
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8
 sp=0xc42003e7e0 pc=0x44b4d1
created by runtime.init.4
  /usr/local/go/src/runtime/proc.go:227 +0x35

goroutine 3 [GC sweep wait]:
runtime.gopark(0x4739b8, 0x4ad7e0, 0x46fdd2, 0xd, 0x419914, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003ef60 sp=0xc42003ef30
 pc=0x42503c
runtime.goparkunlock(0x4ad7e0, 0x46fdd2, 0xd, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003efa0 sp=0xc42003ef60
 pc=0x42512e
runtime.bgsweep(0xc42001e150)
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8
 sp=0xc42003efa0 pc=0x419973
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003efe0
 sp=0xc42003efd8 pc=0x44b4d1
```

```
created by runtime.gcenable
  /usr/local/go/src/runtime/mgc.go:216 +0x58
one more line, no multiline
```

c. Speichern Sie die `test.log`-Datei.

4. Erstellen Sie innerhalb des Ordners `multiline-app` die Docker-Datei.

   a. Fügen Sie den folgenden Inhalt in die Datei ein:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

   b. Speichern Sie die `Dockerfile`-Datei.

5. Erstellen Sie mit der Dockerfile ein Image.

   a. Entwickeln Sie das Image: `docker build -t multiline-app-image .`

      Wobei: `multiline-app-image` der Name für das Image in diesem Beispiel ist.

   b. Überprüfen Sie, ob das Image korrekt erstellt wurde: `docker images –filter reference=multiline-app-image`

      Bei Erfolg zeigt die Ausgabe das Image und das `latest`-Tag.

6. Laden Sie das Image in Amazon-Elastic-Container-Registry hoch.

   a. Erstellen Sie ein Amazon-ECR-Repository zum Speichern des Images: `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Wobei: `multiline-app-repo` der Name für das Repository und `us-east-1` die Region in diesem Beispiel ist.

      Die Ausgabe gibt Ihnen die Details des neuen Repositorys. Notieren Sie sich den `repositoryUri`-Wert, da Sie ihn in den nächsten Schritten benötigen.

b. Markieren Sie Ihr Image mit dem `repositoryUri`-Wert aus der vorherigen Ausgabe: `docker tag multiline-app-image repositoryUri`

   Beispiel: `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Führen Sie das Docker-Image aus, um zu überprüfen, ob es korrekt ausgeführt wurde: `docker images –filter reference=repositoryUri`

   In der Ausgabe ändert sich der Repository-Name von `multiline-app-repo` in den `repositoryUri`-Wert.

d. Verschieben Sie das Image zu Amazon ECR: `docker push aws_account_id.dkr.ecr.region.amazonaws.com/repository name`

   Beispiel: `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

Erstellen einer Aufgabendefinition und Ausführen der Aufgabe

1. Erstellen Sie eine Aufgabendefinitionsdatei mit dem Dateinamen `multiline-task-definition.json`.
2. Fügen Sie den folgenden Inhalt in die `multiline-task-definition.json`-Datei ein:

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN",
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
```

```
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Ersetzen Sie Folgendes in der `multiline-task-definition.json`-Aufgabendefinition:

a. *task role ARN*

   Um den Aufgabenrollen-ARN zu finden, gehen Sie zur IAM-Konsole. Wählen Sie Roles (Rollen) und suchen Sie die `ecs-task-role-for-firelens`-Aufgabenrolle, die Sie erstellt haben. Wählen Sie die Rolle aus und kopieren Sie den ARN, der im Abschnitt Summary (Zusammenfassung) angezeigt wird.

b. *execution role ARN*

   Verwenden Sie die IAM-Konsole, um die Ausführungsrollen-ARN zu finden. Wählen Sie Roles (Rollen) aus und finden Sie die `ecsTaskExecutionRole`-Rolle. Wählen Sie die Rolle aus und kopieren Sie den ARN, der im Abschnitt Summary (Zusammenfassung) angezeigt wird.

c. *aws_account_id*

Um Ihr `aws_account_id` zu finden, melden Sie sich bei AWS Management Console an. Wählen Sie oben rechts Ihren Benutzernamen und kopieren Sie Ihre Konto-ID.

d. *us-east-1*

Ersetzen Sie die Region bei Bedarf.

3. Registrieren Sie die Aufgabendefinitionsdatei: `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region us-east-1`
4. Öffnen Sie die Konsole unter https://console.aws.amazon.com/ecs/v2.
5. Wählen Sie im Navigationsbereich Task Definitions (Aufgabendefinitionen) und dann die `firelens-example-multiline`-Familie aus, da wir die Aufgabendefinition für diese Familie in der ersten Zeile der Aufgabendefinition oben registriert haben.
6. Wählen Sie die neueste Version aus.
7. Wählen Sie Bereitstellen, Aufgabe ausführen aus.
8. Wählen Sie auf der Seite Task ausführen für Cluster den Cluster aus, und wählen Sie dann unter Netzwerk für Subnetze die verfügbaren Subnetze für Ihre Aufgabe aus.
9. Wählen Sie Erstellen.

Stellen Sie sicher, dass mehrzeilige Protokollnachrichten in Amazon verkettet CloudWatch angezeigt werden

1. Öffnen Sie die Konsole unter https://console.aws.amazon.com/cloudwatch/ CloudWatch .
2. Erweitern Sie im Navigationsbereich Logs (Protokolle) und wählen Sie Log groups (Protokollgruppen) aus.
3. Wählen Sie die Protokollgruppe `multiline-test/applicatio` aus.
4. Wählen Sie das Protokoll aus und sehen Sie sich die Nachrichten an. Zeilen, die mit den Regeln in der Parser-Datei übereinstimmen, werden verkettet und werden als einzelne Nachricht angezeigt.

   Das folgende Protokoll-Snippet zeigt einen Go-Stack-Trace, der zu einem einzigen Ereignis verkettet ist:

```
{
    "log": "panic: my panic\n\ngoroutine 4 [running]:\npanic(0x45cb40,
 0x47ad70)\n  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8
 sp=0xc42003f710 pc=0x422f7c\nmain.main.func1(0xc420024120)\n  foo.go:6
```

```
 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0 sp=0xc42003f7d8
 pc=0x44b4d1\ncreated by main.main\n  foo.go:5 +0x58\n\ngoroutine 1 [chan receive]:
\nruntime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)\n  /usr/
local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00 pc=0x42503c
\nruntime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e\nruntime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)\n
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4\nruntime.chanrecv1(0xc420024120, 0x0)\n  /usr/local/go/src/runtime/
chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20 pc=0x40439b\nmain.main()\n
 foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef\nruntime.main()\n
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337
 +0x1 fp=0xc420053fe8 sp=0xc420053fe0 pc=0x44b4d1\n\ngoroutine 2 [force gc
 (idle)]:\nruntime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)\n  /
usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c\nruntime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e\nruntime.forcegchelper()\n  /usr/local/go/src/runtime/proc.go:238
 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8 pc=0x424e5c\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8 sp=0xc42003e7e0
 pc=0x44b4d1\ncreated by runtime.init.4\n  /usr/local/go/src/runtime/proc.go:227
 +0x35\n\ngoroutine 3 [GC sweep wait]:\nruntime.gopark(0x4739b8, 0x4ad7e0,
 0x46fdd2, 0xd, 0x419914, 0x1)\n  /usr/local/go/src/runtime/proc.go:280 +0x12c
 fp=0xc42003ef60 sp=0xc42003ef30 pc=0x42503c\nruntime.goparkunlock(0x4ad7e0,
 0x46fdd2, 0xd, 0x14, 0x1)\n  /usr/local/go/src/runtime/proc.go:286 +0x5e
 fp=0xc42003efa0 sp=0xc42003ef60 pc=0x42512e\nruntime.bgsweep(0xc42001e150)\n
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8 sp=0xc42003efa0
 pc=0x419973\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1
 fp=0xc42003efe0 sp=0xc42003efd8 pc=0x44b4d1\ncreated by runtime.gcenable\n  /usr/
local/go/src/runtime/mgc.go:216 +0x58",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

Das folgende Protokoll-Snippet zeigt, wie dasselbe Ereignis angezeigt wird, wenn Sie einen ECS-Container ausführen, der nicht für die Verkettung von mehrzeiligen Protokollmeldungen konfiguriert ist. Das Protokollfeld enthält eine einzelne Zeile.

```
{
    "log": "panic: my panic",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
```

> **Note**
>
> Wenn Ihre Protokolle anstelle der Standardausgabe in Protokolldateien gehen, empfehlen wir, die `multiline.parser`- und `multiline.key_content`-Konfigurationsparameter im Tail-Eingabe-Plug-In anstelle des Filters anzugeben.

Bereitstellung von Fluent Bit auf Amazon ECS-Windows-Containern

Fluent Bit ist ein schneller und flexibler Protokollprozessor und Router, der von verschiedenen Betriebssystemen unterstützt wird. Es kann verwendet werden, um Protokolle an verschiedene AWS Ziele wie Amazon CloudWatch Logs, Firehose Amazon S3 und Amazon OpenSearch Service weiterzuleiten. Fluent Bit unterstützt gängige Partnerlösungen wie Datadog, Splunk und benutzerdefinierte HTTP-Server. Weitere Informationen zu Fluent Bit finden Sie auf der Fluent Bit -Website.

Das AWS -for-Fluent-Bit-Image ist in Amazon ECR sowohl in der Amazon ECR Public Gallery als auch in einem Amazon-ECR-Repository in den meisten Regionen für hohe Verfügbarkeit verfügbar. Weitere Informationen finden Sie aws-for-fluent-bit auf der GitHub Website.

In diesem Tutorial erfahren Sie, wie Sie Fluent Bit-Container auf ihren Windows-Instances bereitstellen, die in Amazon ECS ausgeführt werden, um von den Windows-Aufgaben generierte Protokolle CloudWatch zur zentralen Protokollierung an Amazon zu streamen.

Dieses Tutorial verwendet den folgenden Ansatz:

- Fluent Bit wird als Service mit der Daemon-Planungstrategie ausgeführt. Diese Strategie stellt sicher, dass eine einzelne Instance von Fluent Bit immer auf den Container-Instances im Cluster ausgeführt wird.

  - Überwacht Port 24224 mithilfe des Weiterleitungs-Eingabe-Plugins.
  - Stellen Sie Port 24224 für den Host bereit, damit die Docker-Laufzeit über diesen bereitgestellten Port Protokolle an Fluent Bit senden kann.
  - Verfügt über eine Konfiguration, die es Fluent Bit ermöglicht, die Protokolldatensätze an bestimmte Ziele zu senden.
- Starten Sie alle anderen Amazon-ECS-Aufgaben-Container mit dem Fluentd-Protokollierungstreiber. Weitere Informationen finden Sie unter Fluentd-Protokollierungstreiber auf der Website der Docker-Dokumentation.
  - Docker stellt eine Verbindung zum TCP-Socket 24224 auf localhost innerhalb des Host-Namespace her.
  - Der Amazon-ECS-Agent fügt den Containern Labels hinzu, die den Cluster-Namen, den Familiennamen der Aufgabendefinition, die Revisionsnummer der Aufgabendefinition, den ARN der Aufgabe und den Containernamen enthalten. Die gleichen Informationen werden dem Protokolldatensatz mithilfe der Labels-Option des Fluentd-Docker-Protokollierungstreibers hinzugefügt. Weitere Informationen finden Sie unter labels, labels-regex, env und env-regex auf der Website der Docker-Dokumentation.
  - Da die `async`-Option des Fluentd-Protokollierungstreibers auf `true` festgelegt ist, puffert Docker beim Neustart des Fluent-Bit-Containers die Protokolle, bis der Fluent-Bit-Container neu gestartet wird. Sie können das Pufferlimit erhöhen, indem Sie die fluentd-buffer-limit Option festlegen. Weitere Informationen finden Sie unter fluentd-buffer-limit auf der Website der Docker-Dokumentation.

Der Arbeitsablauf ist wie folgt:

- Der Fluent-Bit-Container startet und überwacht auf Port 24224, der für den Host bereitgestellt ist.
- Fluent Bit verwendet die in der Aufgabendefinition angegebenen IAM-Rollen-Anmeldeinformationen.
- Andere Aufgaben, die auf derselben Instance gestartet werden, verwenden den Fluentd-Docker-Protokollierungstreiber, um eine Verbindung zum Fluent-Bit-Container auf Port 24224 herzustellen.
- Wenn die Anwendungscontainer Protokolle generieren, markiert die Docker-Laufzeit diese Datensätze, fügt zusätzliche in Labels angegebene Metadaten hinzu und leitet sie dann an Port 24224 im Host-Namespace weiter.
- Fluent Bit empfängt den Protokolldatensatz auf Port 24224, da dieser für den Host-Namespace bereitgestellt ist.

- Fluent Bit führt seine interne Verarbeitung durch und leitet die Protokolle wie angegeben weiter.

In diesem Tutorial wird die CloudWatch Fluent Bit-Standardkonfiguration verwendet, die Folgendes bewirkt:

- Erstellt eine neue Protokollgruppe für jeden Cluster und jede Aufgabendefinitionsfamilie.
- Erstellt einen neuen Protokollstream für jeden Aufgaben-Container in der oben generierten Protokollgruppe, wenn eine neue Aufgabe gestartet wird. Jeder Stream wird mit der Aufgaben-ID gekennzeichnet, zu der der Container gehört.
- Fügt zusätzliche Metadaten hinzu, einschließlich Cluster-Name, Aufgaben-ARN, Aufgaben-Containername, Aufgabendefinitionsfamilie und Revisionsnummer der Aufgabendefinition in jedem Protokolleintrag.

  Wenn Sie beispielsweise with `container_1` and `container_2` und t `task_1` `ask_2` with haben`container_3`, dann sind die folgenden CloudWatch Log-Streams:

  - `/aws/ecs/windows.ecs_task_1`

    `task-out.`*`TASK_ID`*`.container_1`

    `task-out.`*`TASK_ID`*`.container_2`
  - `/aws/ecs/windows.ecs_task_2`

    `task-out.`*`TASK_ID`*`.container_3`

Schritte

- Voraussetzungen
- Schritt 1: Erstellen der IAM-Zugriffsrollen
- Schritt 2: Erstellen der Amazon-ECS-Windows-Container-Instance
- Schritt 3: Konfigurieren von Fluent Bit
- Schritt 4: Registrieren Sie eine Windows Fluent Bit-Aufgabendefinition, an die die Protokolle weitergeleitet werden CloudWatch
- Schritt 5: Ausführen der ecs-windows-fluent-bit-Aufgabendefinition als Amazon-ECS-Service mithilfe der Daemon-Planungsstrategie
- Schritt 6: Registrieren einer Windows-Aufgabendefinition, die die Protokolle generiert
- Schritt 7: Ausführen der windows-app-task-Aufgabendefinition

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.
- Das `aws-for-fluent-bit`-Container-Image ist für die folgenden Windows-Betriebssysteme verfügbar:
  - Windows Server 2019 Core
  - Windows Server 2019 Full
  - Windows Server 2022 Kern
  - Windows Server 2022 Voll
- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Sie verfügen über einen Cluster. In diesem Tutorial lautet der Clustername FluentBit-cluster.
- Sie verfügen über eine VPC mit einem öffentlichen Subnetz, in dem die EC2-Instance gestartet werden soll. Sie können Ihre Standard-VPC verwenden. Sie können auch ein privates Subnetz verwenden, das es CloudWatch Amazon-Endpunkten ermöglicht, das Subnetz zu erreichen. Weitere Informationen zu CloudWatch Amazon-Endpunkten finden Sie unter CloudWatch Amazon-Endpunkte und Kontingente in der. Allgemeine AWS-Referenz Informationen zur Verwendung des Amazon-VPC-Assistenten zum Erstellen einer VPC finden Sie unter the section called “Erstellen einer Virtual Private Cloud”.

Schritt 1: Erstellen der IAM-Zugriffsrollen

Erstellen Sie die Amazon-IAM-Rollen.

1. Erstellen Sie die Amazon ECS-Container-Instance-Rolle mit dem Namen „ecsInstanceRole“. Weitere Informationen finden Sie unter IAM-Rolle der Amazon-ECS-Container-Instance.
2. Erstellen Sie eine IAM-Rolle für die Fluent-Bit-Aufgabe mit dem Namen `fluentTaskRole`. Weitere Informationen finden Sie unter the section called “IAM-Rolle für Aufgabe”.

Die in dieser IAM-Rolle gewährten IAM-Berechtigungen werden von den Aufgaben-Containern übernommen. Damit Fluent Bit Protokolle an senden kann CloudWatch, müssen Sie der Task-IAM-Rolle die folgenden Berechtigungen zuordnen.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:DescribeLogStreams",
            "logs:PutLogEvents"
        ],
        "Resource": "*"
    }
    ]
}
```

3. Fügen Sie der Rolle die -Richtlinie an.

   a. Speichern Sie den obigen Inhalt in einer Datei mit dem Namen `fluent-bit-policy.json`.

   b. Führen Sie den folgenden Befehl aus, um die Inline-Richtlinie der `fluentTaskRole`-IAM-Rolle anzufügen.

   ```
   aws iam put-role-policy --role-name fluentTaskRole --policy-name
    fluentTaskPolicy --policy-document file://fluent-bit-policy.json
   ```

Schritt 2: Erstellen der Amazon-ECS-Windows-Container-Instance

Erstellen Sie eine Windows-Container-Instance für Amazon ECS.

So erstellen Sie eine Amazon-ECS-Instance

1. Verwenden Sie den `aws ssm get-parameters`-Befehl, um die AMI-ID für die Region abzurufen, in der Ihre VPC gehostet wird. Weitere Informationen finden Sie unter Abrufen von ECS-optimierten AMI-Metadaten.

2. Verwenden Sie die Amazon-EC2-Konsole, um die Instance zu starten.

   a. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
   b. Wählen Sie die zu verwendende Region in der Navigationsleiste aus.
   c. Wählen Sie im EC2-Dashboard Launch Instance (Instance starten) aus.
   d. Geben Sie für Name einen eindeutigen Namen ein.
   e. Wählen Sie für Application and OS Images (Amazon Machine Image) (Anwendungs- und Betriebssystem-Images (Amazon-Computer-Image)) das AMI aus, das Sie im ersten Schritt abgerufen haben.
   f. Wählen Sie für Instance type die Option `t3.xlarge` aus.
   g. Wählen Sie für Key pair (login) (Schlüsselpaar (Anmeldung)) ein Schlüsselpaar aus.
   h. Wählen Sie unter Network settings (Netzwerkeinstellungen) für Security group (Sicherheitsgruppe) eine vorhandene Sicherheitsgruppe aus oder erstellen Sie eine neue.
   i. Wählen Sie unter Network settings (Netzwerkeinstellungen), für Auto-assign Public IP (Öffentliche IP automatisch zuweisen), die Option Enable (Aktivieren) aus.
   j. Wählen Sie unter Erweiterte Details für das IAM-Instanzprofil die Option ecs aus. InstanceRole
   k. Konfigurieren Sie Ihre Amazon-ECS-Container-Instance mit den folgenden Benutzerdaten. Fügen Sie unter Advanced Details (Erweiterte Details) das folgende Skript in das Feld User data (Benutzerdaten) ein und ersetzen Sie *`cluster_name`* durch den Namen Ihres Clusters.

      ```
      <powershell>
      Import-Module ECSTools
      Initialize-ECSAgent -Cluster cluster-name -EnableTaskENI -EnableTaskIAMRole -
      LoggingDrivers '["awslogs","fluentd"]'
      </powershell>
      ```

   l. Wenn Sie bereit sind, wählen Sie das Bestätigungsfeld und danach Launch Instances aus.
   m. Auf einer Bestätigungsseite wird Ihnen mitgeteilt, dass die Instance gestartet wird. Wählen Sie View Instances aus, um die Bestätigungsseite zu schließen und zur Konsole zurückzukehren.

Schritt 3: Konfigurieren von Fluent Bit

Sie können die folgende Standardkonfiguration von verwenden AWS , um schnell loszulegen:

- Amazon CloudWatch, das auf dem Fluent Bit-Plug-In für CloudWatchAmazon im offiziellen Fluent Bit-Handbuch basiert.

Alternativ können Sie andere Standardkonfigurationen verwenden, die von bereitgestellt werden. AWS Weitere Informationen finden Sie unter Überschreiben des Einstiegspunkts für das Windows-Image auf der `aws-for-fluent-bit`-Github-Website.

Die Standardkonfiguration von Amazon CloudWatch Fluent Bit ist unten dargestellt.

Ersetzen Sie die folgenden Variablen:

- *`Region`* mit der Region, in die Sie die CloudWatch Amazon-Logs senden möchten.

```
[SERVICE]
    Flush               5
    Log_Level           info
    Daemon              off

[INPUT]
    Name                forward
    Listen              0.0.0.0
    Port                24224
    Buffer_Chunk_Size   1M
    Buffer_Max_Size     6M
    Tag_Prefix          ecs.

# Amazon ECS agent adds the following log keys as labels to the docker container.
# We would use fluentd logging driver to add these to log record while sending it to
 Fluent Bit.
[FILTER]
    Name                modify
    Match               ecs.*
    Rename              com.amazonaws.ecs.cluster ecs_cluster
    Rename              com.amazonaws.ecs.container-name ecs_container_name
    Rename              com.amazonaws.ecs.task-arn ecs_task_arn
    Rename              com.amazonaws.ecs.task-definition-family
 ecs_task_definition_family
    Rename              com.amazonaws.ecs.task-definition-version
 ecs_task_definition_version

[FILTER]
```

```
    Name                rewrite_tag
    Match               ecs.*
    Rule                $ecs_task_arn ^([a-z-:0-9]+)/([a-zA-Z0-9-_]+)/([a-z0-9]+)$
 out.$3.$ecs_container_name false
    Emitter_Name        re_emitted

[OUTPUT]
    Name                cloudwatch_logs
    Match               out.*
    region              region
    log_group_name      fallback-group
    log_group_template  /aws/ecs/$ecs_cluster.$ecs_task_definition_family
    log_stream_prefix   task-
    auto_create_group   On
```

Jedes Protokoll, das in Fluent Bit eingeht, verfügt über ein Tag, das Sie angeben, oder wird automatisch generiert, wenn Sie keins angeben. Die Tags können verwendet werden, um verschiedene Protokolle an verschiedene Ziele weiterzuleiten. Weitere Informationen finden Sie unter Tag im offiziellen Fluent-Bit-Handbuch.

Die oben beschriebene Fluent-Bit-Konfiguration hat die folgenden Eigenschaften:

- Das Weiterleitungs-Eingabe-Plugin überwacht auf eingehendem Datenverkehr auf TCP-Port 24224.
- Jeder an diesem Port empfangene Protokolleintrag verfügt über ein Tag, das das Weiterleitungs-Eingabe-Plugin ändert, um dem Datensatz eine `ecs.`-Zeichenfolge voranzustellen.
- Die interne Fluent-Bit-Pipeline leitet den Protokolleintrag weiter, um den Filter mithilfe des regulären Ausdrucks „Match“ zu ändern. Dieser Filter ersetzt die Schlüssel im JSON-Protokolldatensatz durch das Format, das Fluent Bit verarbeiten kann.
- Der geänderte Protokolleintrag wird dann vom rewrite_tag-Filter verarbeitet. Dieser Filter ändert das Tag des Protokolldatensatzes in das Format out.*TASK_ID*.*CONTAINER_NAME*.
- Das neue Tag wird an das Output-Plug-In cloudwatch_logs weitergeleitet, das die Protokollgruppen und Streams wie zuvor beschrieben mithilfe der `log_stream_prefix` Optionen `log_group_template` und des Ausgabe-Plug-ins erstellt. CloudWatch Weitere Informationen finden Sie unter Konfigurationsparameter im offiziellen Fluent-Bit-Handbuch.

Schritt 4: Registrieren Sie eine Windows Fluent Bit-Aufgabendefinition, an die die Protokolle weitergeleitet werden CloudWatch

Registrieren Sie eine Windows Fluent Bit-Aufgabendefinition, an die die Protokolle weitergeleitet werden. CloudWatch

> **Note**
>
> Diese Aufgabendefinition stellt den Fluent Bit-Container-Port 24224 für den Host-Port 24224 bereit. Stellen Sie sicher, dass dieser Port nicht in Ihrer EC2-Instance-Sicherheitsgruppe geöffnet ist, um externen Zugriff zu verhindern.

So registrieren Sie eine Aufgabendefinition

1. Erstellen Sie eine Datei mit dem Namen `fluent-bit.json` und dem folgenden Inhalt.

   Ersetzen Sie die folgenden Variablen:

   - *`task-iam-role`* durch den Amazon-Ressourcennamen (ARN) Ihrer Aufgaben-IAM-Rolle
   - *`region`* durch die Region, in der Ihre Aufgabe ausgeführt wird

```
{
  "family": "ecs-windows-fluent-bit",
  "taskRoleArn": "task-iam-role",
  "containerDefinitions": [
    {
      "name": "fluent-bit",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-
bit:windowsservercore-latest",
      "cpu": 512,
      "portMappings": [
        {
          "hostPort": 24224,
          "containerPort": 24224,
          "protocol": "tcp"
        }
      ],
      "entryPoint": [
        "Powershell",
```

```
        "-Command"
      ],
      "command": [
        "C:\\entrypoint.ps1 -ConfigFile C:\\ecs_windows_forward_daemon\
\cloudwatch.conf"
      ],
      "environment": [
        {
          "name": "AWS_REGION",
          "value": "region"
        }
      ],
      "memory": 512,
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fluent-bit-logs",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "flb",
          "awslogs-create-group": "true"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Führen Sie den folgenden Befehl aus, um die Aufgabendefinition zu registrieren.

```
aws ecs register-task-definition --cli-input-json file://fluent-bit.json --
region region
```

Sie können die Aufgabendefinitionen für Ihr Konto auflisten, indem Sie den `list-task-definitions`-Befehl ausführen. Die Ausgabe zeigt die Familien- und Revisionswerte an, die Sie zusammen mit `run-task` oder `start-task` verwenden können.

Schritt 5: Ausführen der **`ecs-windows-fluent-bit`**-Aufgabendefinition als Amazon-ECS-Service mithilfe der Daemon-Planungsstrategie

Nachdem Sie eine Aufgabendefinition für Ihr Konto registriert haben, können Sie eine Aufgabe im Cluster ausführen. Für dieses Tutorial führen Sie eine Instance der `ecs-windows-fluent-bit:1`-Aufgabendefinition in Ihrem `FluentBit-cluster`-Cluster aus. Führen Sie die Aufgabe in einem Service aus, der die Daemon-Planungsstrategie verwendet, die sicherstellt, dass auf jeder Ihrer Container-Instances immer eine einzige Instance von Fluent Bit ausgeführt wird.

Ausführen einer Aufgabe

1. Führen Sie den folgenden Befehl aus, um die `ecs-windows-fluent-bit:1`-Aufgabendefinition (im vorherigen Schritt registriert) als Service zu starten.

   > **Note**
   >
   > Diese Aufgabendefinition verwendet den `awslogs`-Protokollierungstreiber. Ihre Container Instance muss über die erforderlichen Berechtigungen verfügen.

   Ersetzen Sie die folgenden Variablen:

   - *`region`* durch die Region, in der Ihr Service ausgeführt wird

   ```
   aws ecs create-service \
       --cluster FluentBit-cluster \
       --service-name FluentBitForwardDaemonService \
       --task-definition ecs-windows-fluent-bit:1 \
       --launch-type EC2 \
       --scheduling-strategy DAEMON \
       --region region
   ```

2. Führen Sie den folgenden Befehl aus, um Ihre Aufgaben aufzulisten.

   Ersetzen Sie die folgenden Variablen:

   - *`region`* durch die Region, in der Ihre Serviceaufgaben ausgeführt werden

```
aws ecs list-tasks --cluster FluentBit-cluster --region region
```

Schritt 6: Registrieren einer Windows-Aufgabendefinition, die die Protokolle generiert

Registrieren Sie eine Aufgabendefinition, die die Protokolle generiert. Diese Aufgabendefinition stellt ein Windows-Container-Image bereit, das jede Sekunde eine inkrementelle Zahl in `stdout` schreibt.

Die Aufgabendefinition verwendet den Fluentd-Protokollierungstreiber, der eine Verbindung zu Port 24224 herstellt, den das Fluent-Bit-Plugin überwacht. Der Amazon-ECS-Agent kennzeichnet jeden Amazon-ECS-Container mit Tags, einschließlich Cluster-Name, Aufgaben-ARN, Familienname der Aufgabendefinition, Revisionsnummer der Aufgabendefinition und Name des Aufgaben-Containers. Diese Schlüsselwert-Labels werden an Fluent Bit übermittelt.

> **Note**
>
> Diese Aufgabe verwendet den `default`-Netzwerkmodus. Sie können jedoch auch den `awsvpc`-Netzwerkmodus mit der Aufgabe verwenden.

So registrieren Sie eine Aufgabendefinition

1. Erstellen Sie eine Datei mit dem Namen `windows-app-task.json` und dem folgenden Inhalt.

```
{
  "family": "windows-app-task",
  "containerDefinitions": [
    {
      "name": "sample-container",
      "image": "mcr.microsoft.com/windows/servercore:ltsc2019",
      "cpu": 512,
      "memory": 512,
      "essential": true,
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
```

```
      "command": [
        "$count=1;while(1) { Write-Host $count; sleep 1; $count=$count+1;}"
      ],
      "logConfiguration": {
        "logDriver": "fluentd",
        "options": {
          "fluentd-address": "localhost:24224",
          "tag": "{{ index .ContainerLabels \"com.amazonaws.ecs.task-definition-
family\" }}",
          "fluentd-async": "true",
          "labels": "com.amazonaws.ecs.cluster,com.amazonaws.ecs.container-
name,com.amazonaws.ecs.task-arn,com.amazonaws.ecs.task-definition-
family,com.amazonaws.ecs.task-definition-version"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Führen Sie den folgenden Befehl aus, um die Aufgabendefinition zu registrieren.

   Ersetzen Sie die folgenden Variablen:

   - *`region`* durch die Region, in der Ihre Aufgabe ausgeführt wird

   ```
   aws ecs register-task-definition --cli-input-json file://windows-app-task.json --
   region region
   ```

   Sie können die Aufgabendefinitionen für Ihr Konto auflisten, indem Sie den `list-task-definitions`-Befehl ausführen. Die Ausgabe zeigt die Familien- und Revisionswerte an, die Sie zusammen mit `run-task` oder `start-task` verwenden können.

Schritt 7: Ausführen der **`windows-app-task`**-Aufgabendefinition

Nachdem Sie die `windows-app-task`-Aufgabendefinition registriert haben, führen Sie sie in Ihrem `FluentBit-cluster`-Cluster aus.

Ausführen einer Aufgabe

1. Führen Sie die `windows-app-task:1`-Aufgabendefinition aus, die Sie im vorherigen Schritt registriert haben.

   Ersetzen Sie die folgenden Variablen:

   - *`region`* durch die Region, in der Ihre Aufgabe ausgeführt wird

   ```
   aws ecs run-task --cluster FluentBit-cluster --task-definition windows-app-task:1
    --count 2 --region region
   ```

2. Führen Sie den folgenden Befehl aus, um Ihre Aufgaben aufzulisten.

   ```
   aws ecs list-tasks --cluster FluentBit-cluster
   ```

Schritt 8: Überprüfen Sie die Anmeldung CloudWatch

Um Ihr Fluent Bit-Setup zu überprüfen, suchen Sie in der CloudWatch Konsole nach den folgenden Protokollgruppen:

- `/ecs/fluent-bit-logs` – Dies ist die Protokollgruppe, die dem Fluent-Bit-Daemon-Container entspricht, der auf der Container-Instance ausgeführt wird.
- `/aws/ecs/FluentBit-cluster.windows-app-task` – Dies ist die Protokollgruppe, die allen Aufgaben entspricht, die für die `windows-app-task`-Aufgabendefinitionsfamilie innerhalb des `FluentBit-cluster`-Clusters gestartet wurden.

  `task-out.`*`FIRST_TASK_ID`*`.sample-container` – Dieser Protokollstream enthält alle Protokolle, die von der ersten Instance der Aufgabe im Beispielcontainer bzw. Aufgaben-Container generiert wurden.

  `task-out.`*`SECOND_TASK_ID`*`.sample-container` – Dieser Protokollstream enthält alle Protokolle, die von der zweiten Instance der Aufgabe im Beispielcontainer bzw. Aufgaben-Container generiert wurden.

Der `task-out.`*`TASK_ID`*`.sample-container`-Protokollstream hat Felder, die den folgenden ähneln:

```
{
    "source": "stdout",
    "ecs_task_arn": "arn:aws:ecs:region:0123456789012:task/FluentBit-
cluster/13EXAMPLE",
    "container_name": "/ecs-windows-app-task-1-sample-container-cEXAMPLE",
    "ecs_cluster": "FluentBit-cluster",
    "ecs_container_name": "sample-container",
    "ecs_task_definition_version": "1",
    "container_id": "61f5e6EXAMPLE",
    "log": "10",
    "ecs_task_definition_family": "windows-app-task"
}
```

So überprüfen Sie die Fluent-Bit-Einrichtung

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Protokollgruppen aus. Stellen Sie sicher, dass Sie sich in der Region befinden, in der Sie Fluent Bit für Ihre Container bereitgestellt haben.

   In der Liste der Protokollgruppen in der AWS-Region sollten Sie Folgendes sehen:

   - `/ecs/fluent-bit-logs`
   - `/aws/ecs/FluentBit-cluster.windows-app-task`

   Wenn Sie diese Protokollgruppen sehen, wurde die Fluent-Bit-Einrichtung verifiziert.

Schritt 9: Bereinigen

Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie die damit verknüpften Ressourcen bereinigen, um zu vermeiden, dass für nicht verwendete Ressourcen Kosten entstehen.

So bereinigen Sie die Tutorial-Ressourcen

1. Halten Sie die `windows-simple-task`-Aufgabe und die `ecs-fluent-bit`-Aufgabe an. Weitere Informationen finden Sie unter the section called “Beenden einer Aufgabe”.
2. Verwenden Sie den folgenden Befehl, um die `/ecs/fluent-bit-logs`-Protokollgruppe zu löschen. Weitere Informationen zum Löschen von Protokollgruppen finden Sie unter delete-log-group in der AWS Command Line Interface -Referenz.

```
aws logs delete-log-group --log-group-name /ecs/fluent-bit-logs
aws logs delete-log-group --log-group-name /aws/ecs/FluentBit-cluster.windows-app-task
```

3. Führen Sie den folgenden Befehl aus, um die Instance zu beenden.

```
aws ec2 terminate-instances --instance-ids instance-id
```

4. Führen Sie den folgenden Befehl aus, um die IAM-Rollen zu löschen.

```
aws iam delete-role --role-name ecsInstanceRole
aws iam delete-role --role-name fluentTaskRole
```

5. Führen Sie den folgenden Befehl aus, um den Amazon-ECS-Cluster zu löschen.

```
aws ecs delete-cluster --cluster FluentBit-cluster
```

Verwendung gMSA für Linux EC2-Container auf Amazon ECS

Amazon ECS unterstützt die Active Directory-Authentifizierung für Linux-Container auf EC2 über ein spezielles Dienstkonto, das als Group Managed Service Account (gMSA) bezeichnet wird.

Linux-basierte Netzwerkanwendungen wie .NET-Core-Anwendungen können Active Directory verwenden, um die Authentifizierung und Autorisierung zwischen Benutzern und Services zu verwalten. Sie können dieses Feature nutzen, indem Sie Anwendungen entwickeln, die mit Active Directory integriert sind und auf Domain-verbundenen Servern laufen. Da Linux-Container jedoch nicht mit einer Domain verbunden werden können, müssen Sie einen Linux-Container für die Ausführung mit gMSA konfigurieren.

Ein Linux-Container, der mit gMSA ausgeführt wird, stützt sich auf den `credentials-fetcher`-Daemon, der auf der Amazon-EC2-Host-Instance des Containers läuft. Das heißt, der Daemon ruft die gMSA-Anmeldeinformationen vom Active Directory-Domain-Controller ab und überträgt diese Anmeldeinformationen an die Container-Instance. Weitere Informationen zu Servicekonten finden Sie unter gMSAs für Windows-Container erstellen auf der Microsoft-Learn-Website.

Überlegungen

Beachten Sie Folgendes, bevor Sie gMSA für Linux-Container verwenden:

- Wenn Ihre Container auf EC2 ausgeführt werden, können Sie sie gMSA für Windows-Container und Linux-Container verwenden. Hinweise gMSA zur Verwendung eines Linux-Containers auf Fargate finden Sie unterWird gMSA für Linux Container auf Fargate verwendet.
- Möglicherweise benötigen Sie einen Windows-Computer, der mit der Domain verbunden ist, um die Voraussetzungen zu erfüllen. Beispielsweise benötigen Sie möglicherweise einen Windows-Computer, welcher mit der Domain verbunden ist, um die gMSA in Active Directory mit PowerShell zu erstellen. Die RSAT Active PowerShell Director-Tools sind nur für Windows verfügbar. Weitere Informationen finden Sie unter Installieren der Active-Directory-Verwaltungstools.
- Sie haben zwischen domainlose gMSA und Hinzufügen jeder Instance zu einer einzelnen Domain gewählt. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

  Wählen Sie dann den Datenspeicher für CredSpec und optional für die Active-Directory-Benutzeranmeldeinformationen für domainlose gMSA.

  Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Container weiterzuleiten. Sie generieren die CredSpec-Datei und speichern sie dann in einer der CredSpec-Speicheroptionen in der folgenden Tabelle, die für das Betriebssystem der Container-Instances spezifisch sind. Um die domainlose Methode zu verwenden, kann ein optionaler Abschnitt in der CredSpec-Datei Anmeldeinformationen in einer der domainless user credentials-Speicheroptionen in der folgenden Tabelle angeben, die für das Betriebssystem der Container-Instances spezifisch sind.

  gMSA-Datenspeicheroptionen nach Betriebssystem

Speicherort	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	Domainlose Benutzer-Anmeldeinformationen	Domainlose Benutzer-Anmeldeinformationen

Speicherort	Linux	Windows
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, domainlose Benutzer-Anmeldeinformationen
Lokale Datei	N/A	CredSpec

Voraussetzungen

Bevor Sie das gMSA-Feature für Linux-Container mit Amazon ECS verwenden, müssen Sie folgende Schritte ausführen:

- Sie richten eine Active-Directory-Domain mit den Ressourcen ein, auf die Ihre Container zugreifen sollen. Amazon ECS unterstützt die folgenden Einrichtungen:
  - Ein AWS Directory Service Active Directory. AWS Directory Service ist ein AWS verwaltetes Active Directory, das auf Amazon EC2 gehostet wird. Weitere Informationen finden Sie unter Erste Schritte mit AWS Managed Microsoft AD im AWS Directory Service Administratorhandbuch.
  - Ein On-Premises-Active-Directory. Sie müssen sicherstellen, dass die Container-Instance von Amazon ECS Linux der Domain beitreten kann. Weitere Informationen finden Sie unter AWS Direct Connect.
- Sie haben ein bestehendes gMSA-Konto im Active Directory. Weitere Informationen finden Sie unter Verwendung gMSA für Linux EC2-Container auf Amazon ECS.
- Sie haben den `credentials-fetcher`-Daemon auf einer Container-Instance von Amazon ECS Linux installiert und führen ihn dort aus. Sie haben dem `credentials-fetcher`-Daemon auch einen ersten Satz von Anmeldeinformationen hinzugefügt, um sich mit dem Active Directory zu authentifizieren.

> **Note**
>
> Der `credentials-fetcher`-Daemon ist nur für Amazon Linux 2023 und Fedora 37 und höher verfügbar. Der Daemon ist für Amazon Linux 2 nicht verfügbar. Weitere Informationen finden Sie unter aws/credentials fetcher on. GitHub

- Sie richten die Anmeldeinformationen für den `credentials-fetcher`-Daemon ein, um sich mit dem Active Directory zu authentifizieren. Die Anmeldeinformationen müssen Mitglied der Active-Directory-Sicherheitsgruppe sein, die Zugriff auf das gMSA-Konto hat. Es gibt mehrere Optionen in Entscheiden Sie, ob Sie die Instances mit der Domain verbinden oder domainlose gMSA verwenden möchten..
- Sie haben die erforderlichen IAM-Berechtigungen hinzugefügt. Welche Berechtigungen erforderlich sind, hängt von den Methoden ab, die Sie für die anfänglichen Anmeldeinformationen und für die Speicherung der Anmeldeinformation wählen:
  - Wenn Sie Domainless gMSA für anfängliche Anmeldeinformationen verwenden, sind IAM-Berechtigungen für für die Aufgabenausführungsrolle erforderlich. AWS Secrets Manager
  - Wenn Sie die Spezifikation der Anmeldeinformationen im SSM Parameter Store speichern, sind IAM-Berechtigungen für den Amazon EC2 Systems Manager Parameter Store für die Rolle der Aufgabenausführung erforderlich.
  - Wenn Sie die Spezifikation der Anmeldeinformationen in Amazon S3 speichern, sind IAM-Berechtigungen für Amazon Simple Storage Service für die Rolle der Aufgabenausführung erforderlich.

Einrichten von gMSA-fähigen Linux-Containern unter Amazon ECS

Die Infrastruktur vorbereiten

Bei den folgenden Schritten handelt es sich um Überlegungen und Einstellungen, die einmal durchgeführt werden müssen. Nachdem Sie diese Schritte ausgeführt haben, können Sie die Erstellung von Container-Instances automatisieren, um diese Konfiguration wiederzuverwenden.

Entscheiden Sie, wie die anfänglichen Anmeldeinformationen bereitgestellt werden, und konfigurieren Sie die EC2-Benutzerdaten in einer wiederverwendbaren EC2-Startvorlage, um den `credentials-fetcher`-Daemon zu installieren.

1. Entscheiden Sie, ob Sie die Instances mit der Domain verbinden oder domainlose gMSA verwenden möchten.
   - EC2-Instances mit der Active Directory-Domain verbinden

- Die Instances anhand von Benutzerdaten hinzufügen

  Fügen Sie die Schritte zum Verbinden der Active-Directory-Domain mit Ihren EC2-Benutzerdaten in eine EC2-Startvorlage ein. Mehrere Gruppen von Amazon EC2 Auto Scaling können dieselbe Startvorlage verwenden.

  Sie können diese Schritte in den Fedora Docs zum Beitreten zu einem Active Directory oder einer FreeIPA-Domain verwenden.

- Erstellen Sie einen Active-Directory-Benutzer für domainlose gMSA

  Der *`credentials-fetcher`*-Daemon verfügt über ein Feature, die als domainlose gMSA bezeichnet wird. Für dieses Feature ist eine Domain erforderlich, die EC2-Instance muss jedoch nicht mit der Domain verbunden werden. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden. Stattdessen geben Sie den Namen eines Secrets in AWS Secrets Manager in der CredSpec-Datei an. Das Secret muss einen Benutzernamen, ein Passwort und die Domain enthalten, bei der die Anmeldung erfolgen soll.

  Dieses Feature wird unterstützt und kann mit Linux- und Windows-Containern verwendet werden.

  Dieses Feature ähnelt dem gMSA support for non-domain-joined container hosts-Feature. Weitere Informationen zum Windows-Feature finden Sie unter gMSA-Architektur und -Verbesserungen auf der Microsoft-Learn-Website.

  a. Erstellen Sie einen Benutzer in Ihrer Active-Directory-Domain. Der Benutzer in Active Directory muss berechtigt sein, auf die gMSA-Servicekonten zuzugreifen, die Sie für die Aufgaben verwenden.

  b. Erstellen Sie ein Geheimnis in AWS Secrets Manager, nachdem Sie den Benutzer in Active Directory erstellt haben. Weitere Informationen finden Sie unter Create an AWS Secrets Manager Secret.

  c. Geben Sie den Benutzernamen, das Passwort und die Domain des Benutzers in JSON-Schlüssel-Wert-Paare mit den Bezeichnungen `username`, `password` bzw. `domainName`, ein.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

d. Fügen Sie der CredSpec-Datei die Konfiguration für das Servicekonto hinzu. Der zusätzliche `HostAccountConfig` enthält den Amazon-Ressourcennamen (ARN) des Secrets in Secrets Manager.

   Auf Windows muss der `PluginGUID` mit der GUID im folgenden Beispielausschnitt übereinstimmen. Auf Linux wird der `PluginGUID` ignoriert. Ersetzen Sie `MySecret` zum Beispiel durch den Amazon-Ressourcennamen (ARN) Ihres Secrets.

```
    "ActiveDirectoryConfig": {
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
```

e. Für das Feature domainlose gMSA sind zusätzliche Berechtigungen in der Aufgabenausführungsrolle erforderlich. Folgen Sie dem Schritt (Optional) domainloses gMSA-Secret.

2. Instances und installieren Sie den **`credentials-fetcher`**-Daemon konfigurieren

   Sie können den `credentials-fetcher`-Daemon mit einem Benutzerdatenskript in Ihrer EC2-Startvorlage installieren. Die folgenden Beispiele veranschaulichen zwei Arten von Benutzerdaten `cloud-config` YAML oder bash-Skript. Diese Beispiele gelten für Amazon Linux 2023 (AL2023). Ersetzen Sie `MyCluster` durch den Namen des Amazon-ECS-Clusters, mit dem diese Instances verbunden werden sollen.

   - **`cloud-config`** YAML

```
Content-Type: text/cloud-config
package_reboot_if_required: true
packages:
  # prerequisites
  - dotnet
  - realmd
  - oddjob
```

```
  - oddjob-mkhomedir
  - sssd
  - adcli
  - krb5-workstation
  - samba-common-tools
  # https://github.com/aws/credentials-fetcher gMSA credentials management for
 containers
  - credentials-fetcher
write_files:
# configure the ECS Agent to join your cluster.
# replace MyCluster with the name of your cluster.
- path: /etc/ecs/ecs.config
  owner: root:root
  permissions: '0644'
  content: |
    ECS_CLUSTER=MyCluster
    ECS_GMSA_SUPPORTED=true
runcmd:
# start the credentials-fetcher daemon and if it succeeded, make it start after
 every reboot
- "systemctl start credentials-fetcher"
- "systemctl is-active credentials-fetch && systemctl enable credentials-
fetcher"
```

- bash-Skript

  Wenn Sie sich mit bash-Skripten besser auskennen und mehrere Variablen in `/etc/ecs/ecs.config` zu schreiben haben, verwenden Sie das folgende `heredoc`-Format. Bei Verwendung dieses Formats wird alles zwischen den Zeilen cat und `EOF` in die Konfigurationsdatei geschrieben.

```
#!/usr/bin/env bash
set -euxo pipefail

# prerequisites
timeout 30 dnf install -y dotnet realmd oddjob oddjob-mkhomedir sssd adcli
 krb5-workstation samba-common-tools
# install https://github.com/aws/credentials-fetcher gMSA credentials
 management for containers
timeout 30 dnf install -y credentials-fetcher

# start credentials-fetcher
```

```
systemctl start credentials-fetcher
systemctl is-active credentials-fetch && systemctl enable credentials-fetcher

cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_GMSA_SUPPORTED=true
EOF
```

Es gibt optionale Konfigurationsvariablen für den `credentials-fetcher`-Daemon, die Sie in `/etc/ecs/ecs.config` einrichten können. Wir empfehlen, dass Sie die Variablen in den Benutzerdaten im YAML-Block oder `heredoc` ähnlich wie in den vorherigen Beispielen festlegen. Dadurch werden Probleme mit der teilweisen Konfiguration vermieden, die beim mehrfachen Bearbeiten einer Datei auftreten können. Weitere Informationen zur Konfiguration des ECS-Agenten finden Sie unter Amazon ECS Container Agent unter GitHub.

- Optional können Sie die Variable `CREDENTIALS_FETCHER_HOST` verwenden, wenn Sie die `credentials-fetcher`-Daemon-Konfiguration ändern, um den Socket an einen anderen Ort zu verschieben.

Einrichten von Berechtigungen und Secrets

Führen Sie die folgenden Schritte einmal für jede Anwendung und jede Aufgabendefinition aus. Wir empfehlen Ihnen, die bewährte Methode anzuwenden, die geringste Berechtigung zu gewähren und die in der Richtlinie verwendeten Berechtigungen einzuschränken. Auf diese Weise kann jede Aufgabe nur die Secrets lesen, die sie benötigt.

1. (Optional) domainloses gMSA-Secret

   Wenn Sie die domainlose Methode verwenden, bei der die Instance nicht mit der Domain verknüpft ist, gehen Sie wie folgt vor.

   Sie müssen die folgenden Berechtigungen als eingebundene Richtlinie zu der IAM-Rolle für die Aufgabenausführung hinzufügen. Dadurch erhält der `credentials-fetcher`-Daemon Zugriff auf das Secrets-Manager-Secret. Ersetzen Sie das Beispiel-`MySecret` durch den Amazon-Ressourcenname (ARN) Ihres Secrets in der `Resource`-Liste.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
   ```

```
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:ssm:aws-region:111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Wenn Sie Ihren eigenen KMS-Schlüssel verwenden, um Ihr Geheimnis zu verschlüsseln, müssen Sie dieser Rolle die erforderlichen Berechtigungen hinzufügen und diese Rolle der AWS KMS Schlüsselrichtlinie hinzufügen.

2. Entscheiden Sie, ob Sie SSM Parameter Store oder S3 zum Speichern von CredSpec verwenden

   Amazon ECS unterstützt die folgenden Möglichkeiten, um auf den Dateipfad im `credentialSpecs`-Feld der Aufgabendefinition zu verweisen.

   Wenn Sie die Instances mit einer einzigen Domain verbinden, verwenden Sie das Präfix `credentialspec:` am Anfang des ARN in der Zeichenfolge. Wenn Sie domainlose gMSA verwenden, verwenden Sie `credentialspecdomainless:`.

   Weitere Informationen zu CredSpecfinden Sie unter Anmeldeinformationsspezifikationsdatei.

   - Amazon S3 Bucket

     Fügen Sie die Anmeldeinformationsspezifikation zu einem Amazon-S3-Bucket hinzu. Dann verweisen Sie auf den Amazon-Ressourcennamen (ARN) des Amazon-S3-Buckets im `credentialSpecs`-Feld der Aufgabendefinition.

     ```
     {
         "family": "",
         "executionRoleArn": "",
         "containerDefinitions": [
             {
                 "name": "",
     ```

```
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/
${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Um Ihren Aufgaben den Zugriff auf den S3-Bucket zu ermöglichen, fügen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur IAM-Rolle für die Aufgabenausführung von Amazon ECS hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/{object}"
            ]
        }
    ]
}
```

- SSM-Parameterspeicher-Parameter

  Fügen Sie die Anmeldeinformationsspezifikation zu einem SSM-Parameter-Store-Parameter hinzu. Dann verweisen Sie auf den Amazon-Ressourcennamen (ARN) des SSM-Parameter-Store-Parameters im `credentialSpecs`-Feld der Aufgabendefinition.

  ```
  {
      "family": "",
      "executionRoleArn": "",
  ```

```
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:ssm:aws-
region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Um Ihren Aufgaben Zugriff auf den Parameter des SSM Parameter Store zu geben, fügen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur Amazon-ECS-IAM-Rolle für die Aufgabenausführung hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:aws-region:111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Anmeldeinformationsspezifikationsdatei

Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Linux-Container weiterzuleiten. Sie erstellen CredSpec und verweisen darauf in dem `credentialSpecs`-Feld in Ihrer Aufgabendefinition. Die CredSpec-Datei enthält keine Secrets.

Im Folgenden sehen Sie ein Beispiel für eine CredSpec-Datei.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Erstellen einer CredSpec

Sie erstellen eine CredSpec indem Sie das Modul CredSpec PowerShell auf einem Windows-Computer verwenden, der mit der Domain verbunden ist. Folgen Sie den Schritten unter Anmeldeinformationsspezifikation erstellen auf der Microsoft-Learn-Website.

Wird gMSA für Linux Container auf Fargate verwendet

Amazon ECS unterstützt die Active Directory-Authentifizierung für Linux-Container auf Fargate über ein spezielles Dienstkonto, das als Group Managed Service Account (gMSA) bezeichnet wird.

Linux-basierte Netzwerkanwendungen wie .NET-Core-Anwendungen können Active Directory verwenden, um die Authentifizierung und Autorisierung zwischen Benutzern und Services zu verwalten. Sie können dieses Feature nutzen, indem Sie Anwendungen entwickeln, die mit Active Directory integriert sind und auf Domain-verbundenen Servern laufen. Da Linux-Container jedoch nicht mit einer Domain verbunden werden können, müssen Sie einen Linux-Container für die Ausführung mit gMSA konfigurieren.

Überlegungen

Beachten Sie Folgendes, bevor Sie es gMSA für Linux Container auf Fargate verwenden:

- Sie müssen die Plattformversion 1.4 oder höher ausführen.
- Möglicherweise benötigen Sie einen Windows-Computer, der mit der Domain verbunden ist, um die Voraussetzungen zu erfüllen. Beispielsweise benötigen Sie möglicherweise einen Windows-Computer, welcher mit der Domain verbunden ist, um die gMSA in Active Directory mit PowerShell zu erstellen. Die RSAT Active PowerShell Director-Tools sind nur für verfügbarWindows. Weitere Informationen finden Sie unter Installieren der Active-Directory-Verwaltungstools.
- Sie müssen Domainless gMSA verwenden.

  Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Container weiterzuleiten. Sie generieren die CredSpec Datei und speichern sie dann in einem Amazon S3 S3-Bucket.
- Eine Aufgabe kann nur ein Active Directory unterstützen.

Voraussetzungen

Bevor Sie das gMSA-Feature für Linux-Container mit Amazon ECS verwenden, müssen Sie folgende Schritte ausführen:

- Sie richten eine Active-Directory-Domain mit den Ressourcen ein, auf die Ihre Container zugreifen sollen. Amazon ECS unterstützt die folgenden Einrichtungen:
  - Ein AWS Directory Service Active Directory. AWS Directory Service ist ein AWS verwaltetes Active Directory, das auf Amazon EC2 gehostet wird. Weitere Informationen finden Sie unter Erste Schritte mit AWS Managed Microsoft AD im AWS Directory Service Administratorhandbuch.

  - Ein On-Premises-Active-Directory. Sie müssen sicherstellen, dass die Container-Instance von Amazon ECS Linux der Domain beitreten kann. Weitere Informationen finden Sie unter AWS Direct Connect.
- Sie haben bereits ein gMSA Konto im Active Directory und einen Benutzer, der berechtigt ist, auf das gMSA Dienstkonto zuzugreifen. Weitere Informationen finden Sie unter Erstellen Sie einen Active-Directory-Benutzer für domainlose gMSA.
- Sie haben einen Amazon S3 S3-Bucket. Weitere Informationen finden Sie unter Erstellen eines Buckets im Amazon S3 S3-Benutzerhandbuch.

Einrichten von gMSA-fähigen Linux-Containern unter Amazon ECS

Die Infrastruktur vorbereiten

Bei den folgenden Schritten handelt es sich um Überlegungen und Einstellungen, die einmal durchgeführt werden müssen.

- Erstellen Sie einen Active-Directory-Benutzer für domainlose gMSA

  Wenn Sie Domainless verwendengMSA, ist der Container nicht mit der Domain verbunden. Andere Anwendungen, die auf dem Container ausgeführt werden, können die Anmeldeinformationen nicht für den Zugriff auf die Domäne verwenden. Aufgaben, die eine andere Domain verwenden, können auf demselben Container ausgeführt werden. Sie geben den Namen eines Geheimnisses AWS Secrets Manager in der CredSpec Datei an. Das Secret muss einen Benutzernamen, ein Passwort und die Domain enthalten, bei der die Anmeldung erfolgen soll.

  Dieses Feature ähnelt dem gMSA support for non-domain-joined container hosts-Feature. Weitere Informationen zum Windows-Feature finden Sie unter gMSA-Architektur und -Verbesserungen auf der Microsoft-Learn-Website.

  a. Konfigurieren Sie einen Benutzer in Ihrer Active Directory-Domäne. Der Benutzer im Active Directory muss berechtigt sein, auf das gMSA Dienstkonto zuzugreifen, das Sie für die Aufgaben verwenden.

  b. Sie haben eine VPC und Subnetze, die den Active Directory-Domainnamen auflösen können. Konfigurieren Sie die VPC mit DHCP-Optionen mit dem Domänennamen, der auf den Active Directory-Dienstnamen verweist. Informationen zur Konfiguration von DHCP-

Optionen für eine VPC finden Sie unter Arbeiten mit DHCP-Optionssätzen im Amazon Virtual Private Cloud Cloud-Benutzerhandbuch.

c. Erstellen Sie ein Geheimnis in. AWS Secrets Manager

d. Erstellen Sie die Datei mit den Anmeldeinformationen.

Einrichten von Berechtigungen und Secrets

Führen Sie die folgenden Schritte einmal für jede Anwendung und jede Aufgabendefinition aus. Wir empfehlen Ihnen, die bewährte Methode anzuwenden, die geringste Berechtigung zu gewähren und die in der Richtlinie verwendeten Berechtigungen einzuschränken. Auf diese Weise kann jede Aufgabe nur die Secrets lesen, die sie benötigt.

1. Erstellen Sie einen Benutzer in Ihrer Active-Directory-Domain. Der Benutzer in Active Directory muss berechtigt sein, auf die gMSA-Servicekonten zuzugreifen, die Sie für die Aufgaben verwenden.
2. Nachdem Sie den Active Directory-Benutzer eingerichtet haben, erstellen Sie ein Geheimnis in AWS Secrets Manager. Weitere Informationen finden Sie unter Ein AWS Secrets Manager - Secret erstellen.
3. Geben Sie den Benutzernamen, das Passwort und die Domain des Benutzers in JSON-Schlüssel-Wert-Paare mit den Bezeichnungen `username`, `password` bzw. `domainName`, ein.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

4. Sie müssen die folgenden Berechtigungen als eingebundene Richtlinie zu der IAM-Rolle für die Aufgabenausführung hinzufügen. Dadurch erhält der `credentials-fetcher`-Daemon Zugriff auf das Secrets-Manager-Secret. Ersetzen Sie das Beispiel-`MySecret` durch den Amazon-Ressourcenname (ARN) Ihres Secrets in der `Resource`-Liste.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:aws-region:111122223333:secret:MySecret"
```

```
      ]
    }
  ]
}
```

> **Note**
>
> Wenn Sie Ihren eigenen KMS-Schlüssel verwenden, um Ihr Geheimnis zu verschlüsseln, müssen Sie dieser Rolle die erforderlichen Berechtigungen hinzufügen und diese Rolle der AWS KMS Schlüsselrichtlinie hinzufügen.

5. Fügen Sie die Anmeldeinformationsspezifikation zu einem Amazon-S3-Bucket hinzu. Dann verweisen Sie auf den Amazon-Ressourcennamen (ARN) des Amazon-S3-Buckets im `credentialSpecs`-Feld der Aufgabendefinition.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Um Ihren Aufgaben den Zugriff auf den S3-Bucket zu ermöglichen, fügen Sie die folgenden Berechtigungen als eingebundene Richtlinie zur IAM-Rolle für die Aufgabenausführung von Amazon ECS hinzu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListObject"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Anmeldeinformationsspezifikationsdatei

Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Linux-Container weiterzuleiten. Sie erstellen CredSpec und verweisen darauf in dem `credentialSpecs`-Feld in Ihrer Aufgabendefinition. Die CredSpec-Datei enthält keine Secrets.

Im Folgenden sehen Sie ein Beispiel für eine CredSpec-Datei.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
```

```
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Eine erstellen CredSpec und auf einen Amazon S3 hochladen

Sie erstellen eine CredSpec indem Sie das Modul CredSpec PowerShell auf einem Windows-Computer verwenden, der mit der Domain verbunden ist. Folgen Sie den Schritten unter Anmeldeinformationsspezifikation erstellen auf der Microsoft-Learn-Website.

Nachdem Sie die Datei mit den Anmeldeinformationen erstellt haben, laden Sie sie in einen Amazon S3 S3-Bucket hoch. Kopieren Sie die CredSpec-Datei auf den Computer oder die Umgebung, in der Sie AWS CLI -Befehle ausführen.

Führen Sie den folgenden AWS CLI Befehl aus, CredSpec um das auf Amazon S3 hochzuladen. Ersetzen Sie `MyBucket` durch den Namen Ihres Amazon-S3-Buckets. Sie können die Datei als Objekt in einem beliebigen Bucket und an einem beliebigen Ort speichern, müssen jedoch in der Richtlinie, die Sie der Aufgabenausführungsrolle zuordnen, den Zugriff auf diesen Bucket und diesen Speicherort gewähren.

Verwenden PowerShell Sie für den folgenden Befehl:

```
$ Write-S3Object -BucketName "MyBucket" -Key "ecs-domainless-gmsa-credspec" -File
 "gmsa-cred-spec.json"
```

Der folgende AWS CLI Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://MyBucket/ecs-domainless-gmsa-credspec
```

Verwenden von Amazon ECS-Windows-Containern mit domainless unter gMSA Verwendung von AWS CLI

Die folgende Anleitung zeigt, wie Sie eine Amazon-ECS-Aufgabe erstellen, die einen Windows-Container ausführt, der über Anmeldeinformationen für den Zugriff auf Active Directory mit AWS CLI verfügt. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

Themen

Voraussetzungen

In diesem Tutorial wird davon ausgegangen, dass die folgenden Voraussetzungen erfüllt wurden:

- Die Schritte in Einrichtung für die Verwendung von Amazon ECS wurden ausgeführt.
- Ihr AWS Benutzer verfügt über die erforderlichen Berechtigungen, die im Beispiel für eine AmazonECS_ FullAccess IAM-Richtlinie angegeben sind.
- Die neueste Version von AWS CLI ist installiert und konfiguriert. Weitere Informationen zur Installation oder Aktualisierung von finden Sie AWS CLI unter Installation von AWS Command Line Interface.
- Sie richten eine Active-Directory-Domain mit den Ressourcen ein, auf die Ihre Container zugreifen sollen. Amazon ECS unterstützt die folgenden Einrichtungen:
  - Ein AWS Directory Service Active Directory. AWS Directory Service ist ein AWS verwaltetes Active Directory, das auf Amazon EC2 gehostet wird. Weitere Informationen finden Sie unter Erste Schritte mit AWS Managed Microsoft AD im AWS Directory Service Administratorhandbuch.
  - Ein On-Premises-Active-Directory. Sie müssen sicherstellen, dass die Container-Instance von Amazon ECS Linux der Domain beitreten kann. Weitere Informationen finden Sie unter AWS Direct Connect.
- Sie haben eine VPC und Subnetze, die den Active Directory-Domainnamen auflösen können.
- Sie haben zwischen domainlose gMSA und Hinzufügen jeder Instance zu einer einzelnen Domain gewählt. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

  Wählen Sie dann den Datenspeicher für CredSpec und optional für die Active-Directory-Benutzeranmeldeinformationen für domainlose gMSA.

  Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Container weiterzuleiten. Sie generieren die CredSpec-Datei und speichern sie dann in einer der CredSpec-Speicheroptionen in der folgenden Tabelle, die für das Betriebssystem der Container-Instances spezifisch sind. Um die domainlose Methode zu verwenden, kann ein optionaler Abschnitt in der CredSpec-Datei Anmeldeinformationen in einer der domainless user credentials-Speicheroptionen in der folgenden Tabelle angeben, die für das Betriebssystem der Container-Instances spezifisch sind.

gMSA-Datenspeicheroptionen nach Betriebssystem

Speicherort	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	Domainlose Benutzer-Anmeldeinformationen	Domainlose Benutzer-Anmeldeinformationen
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, domainlose Benutzer-Anmeldeinformationen
Lokale Datei	N/A	CredSpec

- (Optional) AWS CloudShell ist ein Tool, das Kunden eine Befehlszeile bietet, ohne ihre eigene EC2-Instance erstellen zu müssen. Weitere Informationen finden Sie unter Was ist? AWS CloudShell im AWS CloudShell Benutzerhandbuch.

Schritt 1: Das gMSA-Konto in den Active Directory-Domain-Services (AD DS) erstellen und konfigurieren

Erstellen und konfigurieren Sie ein gMSA-Konto in der Active Directory-Domain.

1. Ein Stammschlüssel für den Schlüsselverteilungs-Service generieren

   > **Note**
   >
   > Wenn Sie verwenden AWS Directory Service, können Sie diesen Schritt überspringen.

   Der KDS-Stammschlüssel und die gMSA-Berechtigungen werden mit Ihrem AWS Managed Microsoft AD konfiguriert.

   Wenn Sie in Ihrer Domain noch kein gMSA-Servicekonto erstellt haben, müssen Sie zunächst einen Stammschlüssel für den Key Distribution Service (KDS, Schlüsselverteilung) generieren. KDS ist für die Erstellung, Rotation und Weitergabe des gMSA-Passworts an autorisierte Hosts

verantwortlich. Wenn `ccg.exe` gMSA-Anmeldeinformationen abrufen muss, wendet es sich an KDS, um das aktuelle Passwort abzurufen.

Um zu überprüfen, ob der KDS-Stammschlüssel bereits erstellt wurde, führen Sie mithilfe des Moduls das folgende PowerShell Cmdlet mit Domänenadministratorrechten auf einem Domänencontroller aus. `ActiveDirectory` PowerShell Weitere Informationen zum Modul finden Sie unter ActiveDirectory Modul auf der Microsoft Learn-Website.

```
PS C:\> Get-KdsRootKey
```

Wenn der Befehl eine Schlüssel-ID zurückgibt, können Sie den Rest dieses Schritts überspringen. Erstellen Sie andernfalls den KDS-Stammschlüssel, indem Sie den folgenden Befehl ausführen:

```
PS C:\> Add-KdsRootKey -EffectiveImmediately
```

Obwohl das Argument `EffectiveImmediately` des Befehls impliziert, dass der Schlüssel sofort gültig ist, müssen Sie 10 Stunden warten, bis der KDS-Stammschlüssel repliziert und auf allen Domain-Controllern verwendet werden kann.

2. Das gMSA-Konto erstellen

   Um das gMSA Konto zu erstellen und das Abrufen des gMSA Kennworts `ccg.exe` zu ermöglichen, führen Sie die folgenden PowerShell Befehle von einem Windows-Server oder -Client aus, der Zugriff auf die Domäne hat. Ersetzen Sie `ExampleAccount` durch den Namen, den Sie für Ihr gMSA-Konto verwenden möchten.

   a.
   ```
   PS C:\> Install-WindowsFeature RSAT-AD-PowerShell
   ```

   b.
   ```
   PS C:\> New-ADGroup -Name "ExampleAccount Authorized Hosts" -SamAccountName
    "ExampleAccountHosts" -GroupScope DomainLocal
   ```

   c.
   ```
   PS C:\> New-ADServiceAccount -Name "ExampleAccount" -DnsHostName
    "contoso" -ServicePrincipalNames "host/ExampleAccount", "host/contoso" -
   PrincipalsAllowedToRetrieveManagedPassword "ExampleAccountHosts"
   ```

   d. Erstellen Sie einen Benutzer mit einem permanenten Passwort, das nicht abläuft. Diese Anmeldeinformationen werden in jeder Aufgabe gespeichert AWS Secrets Manager und von ihnen verwendet, um der Domäne beizutreten.

```
PS C:\> New-ADUser -Name "ExampleAccount" -AccountPassword (ConvertTo-
SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleAccount"
```

f. Installieren Sie das PowerShell Modul zum Erstellen von CredSpec Objekten in Active Directory und geben Sie das CredSpec JSON aus.

```
PS C:\> Install-PackageProvider -Name NuGet -Force
```

```
PS C:\> Install-Module CredentialSpec
```

g.
```
PS C:\> New-CredentialSpec -AccountName ExampleAccount
```

3. Kopieren Sie die JSON-Ausgabe des vorherigen Befehls in eine Datei mit dem Namen `gmsa-cred-spec.json`. Das ist die CredSpec-Datei. Sie wird in Schritt 3, Schritt 3: Ihre CredSpec-JSON-Datei so ändern, dass sie Informationen von domainlosen gMSA enthält, verwendet.

Schritt 2: Die Anmeldeinformationen in Secrets Manager hochladen

Kopieren Sie die Active-Directory-Anmeldeinformationen in ein sicheres Speichersystem für Anmeldeinformationen, sodass sie bei jeder Aufgabe abgerufen werden. Dies ist die domainlose gMSA Methode. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

- Führen Sie den folgenden AWS CLI Befehl aus und ersetzen Sie den Benutzernamen, das Passwort und den Domainnamen entsprechend Ihrer Umgebung. Bewahren Sie den ARN des Secrets auf, um es im nächsten Schritt, Schritt 3: Ihre CredSpec-JSON-Datei so ändern, dass sie Informationen von domainlosen gMSA enthält, zu verwenden

Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

```
$ aws secretsmanager create-secret \
--name gmsa-plugin-input \
--description "Amazon ECS - gMSA Portable Identity." \
--secret-string "{\"username\":\"ExampleAccount\",\"password\":\"Test123\",
\"domainName\":\"contoso.com\"}"
```

Schritt 3: Ihre CredSpec-JSON-Datei so ändern, dass sie Informationen von domainlosen gMSA enthält

Bevor Sie das CredSpec in eine der Speicheroptionen hochladen, fügen Sie der CredSpec Informationen mit dem ARN des Secrets im Secrets Manager aus dem vorherigen Schritt hinzu. Weitere Informationen finden Sie unter Konfiguration zusätzlicher Anmeldeinformationen für den Anwendungsfall non-domain-joined Container-Hosts auf der Microsoft Learn-Website.

1. Fügen Sie der CredSpec-Datei in der `ActiveDirectoryConfig` die folgende Information hinzu. Ersetzen Sie den ARN durch das Secret im Secrets Manager aus dem vorherigen Schritt.

   Beachten Sie, dass der `PluginGUID`-Wert mit der GUID im folgenden Beispielausschnitt übereinstimmen muss und erforderlich ist.

   ```
   "HostAccountConfig": {
         "PortableCcgVersion": "1",
         "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
         "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
   region:111122223333:secret:gmsa-plugin-input\"}"
       }
   ```

   Sie können auch ein Secret im SSM Parameter Store verwenden, indem Sie den ARN in diesem Format verwenden: `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

2. Nachdem Sie die CredSpec-Datei geändert haben, sollte sie wie im folgenden Beispiel aussehen:

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "ExampleAccount",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "contoso"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Schritt 4: CredSpec auf Amazon S3 hochladen

In diesem Schritt wird der verwendet. AWS CLI Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Kopieren Sie die CredSpec-Datei auf den Computer oder die Umgebung, in der Sie AWS CLI -Befehle ausführen.

2. Führen Sie den folgenden AWS CLI Befehl aus, CredSpec um das auf Amazon S3 hochzuladen. Ersetzen Sie `MyBucket` durch den Namen Ihres Amazon-S3-Buckets. Sie können die Datei als Objekt in einem beliebigen Bucket und an einem beliebigen Ort speichern, müssen jedoch in der Richtlinie, die Sie der Aufgabenausführungsrolle zuordnen, den Zugriff auf diesen Bucket und diesen Speicherort gewähren.

   Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   $ aws s3 cp gmsa-cred-spec.json \
   s3://MyBucket/ecs-domainless-gmsa-credspec
   ```

Schritt 5 (Optional): Ein Amazon-ECS-Cluster erstellen

Standardmäßig hat Ihr Konto einen Amazon-ECS-Cluster mit dem Namen `default`. Dieser Cluster wird standardmäßig in den AWS CLI SDKs und AWS CloudFormation verwendet. Sie können zusätzliche Cluster verwenden, um Aufgaben und Infrastruktur zu gruppieren und zu organisieren und bestimmten Konfigurationen Standardwerte zuzuweisen.

Sie können einen Cluster aus den AWS Management Console, AWS CLI, SDKs oder erstellen. AWS CloudFormation Die Einstellungen und die Konfiguration im Cluster wirken sich nicht auf gMSA aus.

In diesem Schritt wird der AWS CLI verwendet. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

```
$ aws ecs create-cluster --cluster-name windows-domainless-gmsa-cluster
```

> **Important**
>
> Wenn Sie einen eigenen Cluster erstellen möchten, müssen Sie für jeden Befehl, den Sie mit diesem Cluster verwenden möchten, `--cluster clusterName` angeben.

Schritt 6: Eine IAM-Rolle für Container-Instances erstellen

Eine Container-Instance ist ein Host-Computer zum Ausführen von Containern in ECS-Aufgaben, beispielsweise Amazon-EC2-Instances. Jede Container-Instance wird bei einem Amazon ECS-

Cluster registriert. Bevor Sie Amazon-EC2-Instances starten und sie in einem Cluster registrieren, müssen Sie eine IAM-Rolle erstellen, die ihre Container-Instances verwenden können.

Wie Sie die Rolle der Container-Instance erstellen, erfahren Sie unter IAM-Rolle für Amazon-ECS-Container-Instance. Die Standard-`ecsInstanceRole` verfügt über ausreichende Berechtigungen, um dieses Tutorial abzuschließen.

Schritt 7: Eine benutzerdefinierte Aufgaben-Ausführungsrolle erstellen

Amazon ECS kann anstelle der Container-Instance-Rolle eine andere IAM-Rolle für die Berechtigungen verwenden, die zum Starten der einzelnen Aufgaben erforderlich sind. Diese Rolle ist die Aufgabenausführungsrolle. Wir empfehlen, eine Aufgabenausführungsrolle nur mit den Berechtigungen zu erstellen, die ECS zur Ausführung der Aufgabe benötigt, auch bekannt als geringste Berechtigung. Weitere Informationen zum Prinzip der geringsten Berechtigung finden Sie unter SEC03-BP02 Zugriff mit geringster Berechtigung gewähren im AWS -Well-Architected-Framework.

1. Um eine Aufgabenausführungsrolle zu erstellen, siehe Erstellen der -Aufgabenausführungsrolle. Die Standardberechtigungen ermöglichen es der Container-Instance, Container-Images aus Amazon Elastic Container Registry `stdout` und `stderr` aus Ihren Anwendungen abzurufen, um sie in Amazon CloudWatch Logs zu protokollieren.

   Da die Rolle für dieses Tutorial benutzerdefinierte Berechtigungen benötigt, können Sie der Rolle einen anderen Namen als `ecsTaskExecutionRole` geben. In diesem Tutorial wird `ecsTaskExecutionRole` in weiteren Schritten verwendet.
2. Fügen Sie die folgenden Berechtigungen hinzu, indem Sie eine benutzerdefinierte Richtlinie erstellen, entweder eine Inline-Richtlinie, die nur für diese Rolle existiert, oder eine Richtlinie, die Sie wiederverwenden können. Ersetzen Sie den ARN für die `Resource` in der ersten Anweisung durch den Amazon-S3-Bucket und den Speicherort und die zweite `Resource` durch den ARN des Secrets im Secrets Manager.

   Wenn Sie das Secret im Secrets Manager mit einem benutzerdefinierten Schlüssel verschlüsseln, müssen Sie auch `kms:Decrypt` für den Schlüssel zulassen.

   Wenn Sie SSM Parameter Store anstelle von Secrets Manager verwenden, müssen Sie `ssm:GetParameter` für den Parameter anstelle von `secretsmanager:GetSecretValue` zulassen.

   ```
   {
   ```

```
"Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::MyBucket/ecs-domainless-gmsa-credspec/gmsa-cred-
spec.json"
    },
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": "arn:aws:secretsmanager:aws-region:111122223333:secret:gmsa-
plugin-input"
    }
  ]
}
```

Schritt 8: Eine Aufgabenrolle für Amazon ECS Exec erstellen

In diesem Tutorial wird Amazon ECS Exec verwendet, um die Funktionalität zu überprüfen, indem ein Befehl innerhalb einer laufenden Aufgabe ausgeführt wird. Um ECS Exec verwenden zu können, muss der Service oder die Aufgabe ECS Exec aktivieren und die Aufgabenrolle (aber nicht die Rolle zur Aufgabenausführung) muss über `ssmmessages`-Berechtigungen verfügen. Informationen zur erforderlichen IAM-Richtlinie finden Sie unter ECS Exec-Berechtigungen.

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

Gehen Sie folgendermaßen vor AWS CLI, um eine Aufgabenrolle mithilfe von zu erstellen.

1. Erstellen Sie eine Datei mit dem Namen `ecs-tasks-trust-policy.json` und den folgenden Inhalten:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Erstellen Sie eine IAM-Rolle. Sie können den Namen `ecs-exec-demo-task-role` ersetzen, aber behalten Sie den Namen für die folgenden Schritte bei.

   Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   $ aws iam create-role --role-name ecs-exec-demo-task-role \
   --assume-role-policy-document file://ecs-tasks-trust-policy.json
   ```

   Sie können die Datei `ecs-tasks-trust-policy.json` löschen.

3. Erstellen Sie eine Datei mit dem Namen `ecs-exec-demo-task-role-policy.json` und den folgenden Inhalten:

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "ssmmessages:CreateControlChannel",
                   "ssmmessages:CreateDataChannel",
                   "ssmmessages:OpenControlChannel",
                   "ssmmessages:OpenDataChannel"
               ],
               "Resource": "*"
           }
       ]
   }
   ```

4. Erstellen Sie eine IAM-Richtlinie und verknüpfen Sie sie mit der Rolle aus dem vorherigen Schritt.

Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

```
$ aws iam put-role-policy \
    --role-name ecs-exec-demo-task-role \
    --policy-name ecs-exec-demo-task-role-policy \
    --policy-document file://ecs-exec-demo-task-role-policy.json
```

Sie können die Datei `ecs-exec-demo-task-role-policy.json` löschen.

Schritt 9: Eine Aufgabendefinition erstellen, die domainlose gMSA verwendet

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Erstellen Sie eine Datei mit dem Namen `windows-gmsa-domainless-task-def.json` und den folgenden Inhalten:

```
{
  "family": "windows-gmsa-domainless-task",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "memory": 1024,
      "essential": true,
      "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::ecs-domainless-gmsa-
credspec/gmsa-cred-spec.json"
      ],
      "entryPoint": [
        "powershell",
        "-Command"
      ],
      "command": [
```

```
        "New-Item -Path C:\\inetpub\\wwwroot\\index.html -ItemType file -Value
 '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>' -Force ; C:\
\ServiceMonitor.exe w3svc"
      ],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80,
          "hostPort": 8080
        }
      ]
    }
  ],
  "taskRoleArn": "arn:aws:iam::111122223333:role/ecs-exec-demo-task-role",
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

2. Registrieren Sie die Aufgabendefinition, indem Sie den folgenden Befehl ausführen:

   Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   $ aws ecs register-task-definition \
   --cli-input-json file://windows-gmsa-domainless-task-def.json
   ```

Schritt 10: Eine Windows-Container-Instance im Cluster registrieren

Starten Sie eine Amazon-EC2-Windows-Instance und führen Sie den ECS-Container-Agenten aus, um sie als Container-Instance im Cluster zu registrieren. ECS führt Aufgaben auf den Container-Instances aus, die in dem Cluster registriert sind, in dem die Aufgaben gestartet werden.

1. Informationen zum Starten einer Amazon EC2 EC2-Windows-Instance, die für Amazon ECS in konfiguriert ist AWS Management Console, finden Sie unterStarten einer Amazon ECS Windows-Container-Instance. Halten Sie beim Schritt für Benutzerdaten an.
2. Für gMSA müssen die Benutzerdaten die Umgebungsvariable `ECS_GMSA_SUPPORTED` festlegen, bevor der ECS-Container-Agent gestartet wird.

Für ECS Exec muss der Agent mit dem Argument `-EnableTaskIAMRole` beginnen.

Um die Instance-IAM-Rolle zu sichern, indem verhindert wird, dass Aufgaben den EC2-IMDS-Webservice zum Abrufen der Rollenanmeldedaten erreichen, fügen Sie das Argument `-AwsvpcBlockIMDS` hinzu. Dies gilt nur für Aufgaben, die den `awsvpc`-Netzwerkmodus verwenden.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_GMSA_SUPPORTED", $TRUE, "Machine")
Import-Module ECSTools
Initialize-ECSAgent -Cluster windows-domainless-gmsa-cluster -EnableTaskIAMRole -
AwsvpcBlockIMDS
</powershell>
```

3. Überprüfen Sie Ihre Instance-Konfiguration im Bereich Summary (Übersicht). Wenn alles in Ordnung ist, klicken Sie auf Launch instance (Instance starten).

Schritt 11: Container-Instance überprüfen

Mithilfe der AWS Management Console können Sie überprüfen, ob es im Cluster eine Container-Instance gibt. gMSA benötigt jedoch zusätzliche Features, die als Attribute angegeben sind. Diese Attribute sind in der nicht sichtbar AWS Management Console, daher verwendet dieses Tutorial die AWS CLI.

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Die Container-Instances im Cluster auflisten. Container-Instances haben eine ID, die sich von der ID der EC2-Instance unterscheidet.

```
$ aws ecs list-container-instances
```

Ausgabe:

```
{
    "containerInstanceArns": [
```

```
        "arn:aws:ecs:aws-region:111122223333:container-
instance/default/MyContainerInstanceID"
    ]
}
```

`526bd5d0ced448a788768334e79010fd` ist beispielsweise eine gültige Container-Instance-ID.

2. Verwenden Sie die Container-Instance-ID aus dem vorherigen Schritt, um die Details für die Container-Instance abzurufen. Ersetzen Sie `MyContainerInstanceID` durch die ID.

   Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   $ aws ecs describe-container-instances \
        ----container-instances MyContainerInstanceID
   ```

   Beachten Sie, dass die Ausgabe sehr lang ist.

3. Stellen Sie sicher, dass die `attributes`-Liste ein Objekt mit dem Schlüssel namens `name` und einem Wert `ecs.capability.gmsa-domainless` enthält. Es folgt ein Beispiel für das Objekt.

   Ausgabe:

   ```
   {
       "name": "ecs.capability.gmsa-domainless"
   }
   ```

Schritt 12: Eine Windows-Aufgabe ausführen

Führen Sie eine Amazon-ECS-Aufgabe aus. Wenn es im Cluster nur 1 Container-Instance gibt, können Sie `run-task` verwenden. Wenn es viele verschiedene Container-Instances gibt, kann es einfacher sein, `start-task` zu verwenden und die ID der Container-Instance anzugeben, auf der die Aufgabe ausgeführt werden soll, als der Aufgabendefinition Einschränkungen für die Platzierung hinzuzufügen, um zu steuern, auf welchem Typ von Container-Instance die Aufgabe ausgeführt werden soll.

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   aws ecs run-task --task-definition windows-gmsa-domainless-task \
       --enable-execute-command --cluster windows-domainless-gmsa-cluster
   ```

   Notieren Sie sich die Aufgaben-ID, die von dem Befehl zurückgegeben wird.

2. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob die Aufgabe gestartet wurde. Dieser Befehl wartet und gibt die Shell-Eingabeaufforderung erst zurück, wenn die Aufgabe startet. Ersetzen Sie `MyTaskID` mit der Aufgaben-ID aus dem vorherigen Schritt.

   ```
   $ aws ecs wait tasks-running --task MyTaskID
   ```

Schritt 13: Sicherstellen, dass der Container über gMSA-Anmeldeinformationen verfügt

Stellen Sie sicher, dass der Container in der Aufgabe über ein Kerberos-Token verfügt. gMSA

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

   ```
   $ aws ecs execute-command \
   --task MyTaskID \
   --container windows_sample_app \
   --interactive \
   --command powershell.exe
   ```

   Die Ausgabe erfolgt in einer PowerShell Eingabeaufforderung.

2. Führen Sie den folgenden Befehl im PowerShell Terminal im Container aus.

```
PS C:\> klist get ExampleAccount$
```

Beachten Sie in der Ausgabe, dass der `Principal` der ist, den Sie zuvor erstellt haben.

Schritt 14: Bereinigen

Wenn Sie mit diesem Tutorial fertig sind, sollten Sie die zugehörigen Ressourcen bereinigen, um zu vermeiden, dass Gebühren für ungenutzte Ressourcen anfallen.

In diesem Schritt wird der verwendet AWS CLI. Sie können diese Befehle in AWS CloudShell in der Standard-Shell ausführen, und zwar `bash`.

1. Die Aufgabe anhalten. Ersetzen Sie `MyTaskID` durch die Aufgaben-ID aus Schritt 12, Schritt 12: Eine Windows-Aufgabe ausführen.

   ```
   $ aws ecs stop-task --task MyTaskID
   ```

2. Beenden Sie die Amazon-EC2-Instance. Danach wird die Container-Instance im Cluster nach einer Stunde automatisch gelöscht.

   Sie können die Instance über die Amazon-EC2-Konsole finden und beenden. Sie können alternativ den folgenden Befehl ausführen. Um den Befehl auszuführen, suchen Sie die EC2-Instance-ID in der Ausgabe des `aws ecs describe-container-instances`-Befehls aus Schritt 1, Schritt 11: Container-Instance überprüfen. i-10a64379 ist ein Beispiel für eine EC2-Instance-ID.

   ```
   $ aws ec2 terminate-instances --instance-ids MyInstanceID
   ```

3. Löschen Sie die CredSpec-Datei in Amazon S3. Ersetzen Sie `MyBucket` durch den Namen Ihres Amazon-S3-Buckets.

   ```
   $ aws s3api delete-object --bucket MyBucket --key ecs-domainless-gmsa-credspec/
   gmsa-cred-spec.json
   ```

4. Das Secret aus Secrets Manager löschen. Wenn Sie stattdessen SSM Parameter Store verwendet haben, löschen Sie den Parameter.

Der folgende Befehl verwendet Backslash-Fortsetzungszeichen, die von `sh` und kompatiblen Shells verwendet werden. Dieser Befehl ist nicht kompatibel mit PowerShell. Sie müssen den Befehl ändern, um ihn mit verwenden zu können PowerShell.

```
$ aws secretsmanager delete-secret --secret-id gmsa-plugin-input \
    --force-delete-without-recovery
```

5. Melden Sie die Aufgabendefinition ab und löschen Sie sie. Wenn Sie die Registrierung der Aufgabendefinition aufheben, markieren Sie sie als inaktiv, sodass sie nicht zum Starten neuer Aufgaben verwendet werden kann. Anschließend können Sie die Aufgabendefinition löschen.

   a. Heben Sie die Registrierung der Aufgabendefinition auf, indem Sie die Version angeben. ECS erstellt automatisch Versionen von Aufgabendefinitionen, die von 1 an nummeriert sind. Sie verweisen auf die Versionen im gleichen Format wie die Beschriftungen auf Container-Images, z. B. `:1`.

   ```
   $ aws ecs deregister-task-definition --task-definition windows-gmsa-domainless-task:1
   ```

   b. Löschen Sie die Aufgabendefinition.

   ```
   $ aws ecs delete-task-definitions --task-definition windows-gmsa-domainless-task:1
   ```

6. (Optional) Löschen Sie den ECS-Cluster, falls Sie einen Cluster erstellt haben.

```
$ aws ecs delete-cluster --cluster windows-domainless-gmsa-cluster
```

Debuggen von Amazon ECS domainlosen gMSA für Windows-Container

Amazon-ECS-Aufgabenstatus

ECS versucht, eine Aufgabe genau einmal zu starten. Jede Aufgabe, bei der ein Problem auftritt, wird angehalten und auf den Status `STOPPED` gesetzt. Es gibt zwei häufige Arten von Problemen mit Aufgaben. Erstens Aufgaben, die nicht gestartet werden konnten. Zweitens Aufgaben, bei denen die Anwendung in einem der Container gestoppt wurde. Suchen Sie im AWS Management Console Feld Grund für den Stopp der Aufgabe nach dem Grund, warum die Aufgabe gestoppt

wurde. In der AWS CLI, beschreiben Sie die Aufgabe und sehen Sie sich den `stoppedReason` an. Die Schritte im Feld AWS Management Console und AWS CLI finden Sie unterFehler beim Beenden von Amazon ECS-Aufgaben anzeigen.

Windows-Ereignisse

Windows-Ereignisse für gMSA in Containern werden in der `Microsoft-Windows-Containers-CCG`-Protokolldatei protokolliert und befinden sich in der Ereignisanzeige im Abschnitt Anwendungen und Services in `Logs\Microsoft\Windows\Containers-CCG\Admin`. Weitere Tipps zum Debuggen finden Sie unter Problembehandlung bei gMSAs für Windows-Container auf der Microsoft-Learn-Website.

ECS-Agent-gMSA-Plug-In

Die Protokollierung des gMSA-Plug-ins für den ECS-Agenten auf der Windows-Container-Instance befindet sich im folgenden Verzeichnis, `C:/ProgramData/Amazon/gmsa-plugin/`. Sehen Sie in diesem Protokoll nach, ob die domainlosen Benutzeranmeldeinformationen vom Speicherort, z. B. Secrets Manager, heruntergeladen wurden und ob das Format der Anmeldeinformationen korrekt gelesen wurde.

Erfahren Sie, wie Sie GMSAs für EC2-Windows-Container für Amazon ECS verwenden

Amazon ECS unterstützt die Active Directory-Authentifizierung für Windows-Container über ein spezielles Service-Konto namens group Managed Service Account (gMSA).

Windows-basierte Netzwerkanwendungen wie .NET-Anwendungen verwenden oft Active Directory, um die Authentifizierung und das Autorisierungsmanagement zwischen Benutzern und Services zu erleichtern. Entwickler entwerfen ihre Anwendungen häufig so, dass sie sich in Active Directory integrieren lassen und zu diesem Zweck auf der Domain angehörenden Servern ausgeführt werden. Da Windows-Container nicht mit einer Domain verbunden werden können, müssen Sie einen Windows-Container für die Ausführung mit dem gMSA konfigurieren.

Ein Windows-Container, der mit dem gMSA ausgeführt wird, ist zur Abfrage der gMSA-Anmeldeinformationen vom Active Directory-Domain-Controller und zur Bereitstellung an die Container-Instance auf die Amazon-EC2-Instance seines Hosts angewiesen. Weitere Informationen finden Sie unter gMSAs für Windows-Container erstellen.

> **Note**
>
> Für Windows-Container in Fargate wird dieses Feature nicht unterstützt.

Themen

Überlegungen

Die folgenden Punkte sollten bei der Verwendung von gMSAs für Windows-Container berücksichtigt werden:

- Wenn Sie das für Amazon-ECS-optimierte Windows Server 2016 Full AMI für Ihre Container-Instances verwenden, muss der Container-Hostname mit dem gMSA-Kontonamen übereinstimmen, der in der Anmeldeinformations-Spezifikationsdatei definiert ist. Um einen Hostnamen für einen Container anzugeben, verwenden Sie den `hostname`-Containerdefinitionsparameter. Weitere Informationen finden Sie unter Network settings (Netzwerkeinstellungen).
- Sie haben zwischen domainlose gMSA und Hinzufügen jeder Instance zu einer einzelnen Domain gewählt. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

  Wählen Sie dann den Datenspeicher für CredSpec und optional für die Active-Directory-Benutzeranmeldeinformationen für domainlose gMSA.

  Amazon ECS verwendet eine Spezifikationsdatei für Active-Directory-Anmeldeinformationen (CredSpec). Diese Datei enthält die gMSA-Metadaten, die verwendet werden, um den gMSA-Kontokontext an den Container weiterzuleiten. Sie generieren die CredSpec-Datei und speichern sie dann in einer der CredSpec-Speicheroptionen in der folgenden Tabelle, die für das Betriebssystem der Container-Instances spezifisch sind. Um die domainlose Methode zu verwenden, kann ein optionaler Abschnitt in der CredSpec-Datei Anmeldeinformationen in einer

der domainless user credentials-Speicheroptionen in der folgenden Tabelle angeben, die für das Betriebssystem der Container-Instances spezifisch sind.

gMSA-Datenspeicheroptionen nach Betriebssystem

Speicherort	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	Domainlose Benutzer-Anmeldeinformationen	Domainlose Benutzer-Anmeldeinformationen
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, domainlose Benutzer-Anmeldeinformationen
Lokale Datei	N/A	CredSpec

Voraussetzungen

Bevor Sie das gMSA-Feature für Windows-Container mit Amazon ECS verwenden, müssen Sie folgende Schritte ausführen:

- Sie richten eine Active-Directory-Domain mit den Ressourcen ein, auf die Ihre Container zugreifen sollen. Amazon ECS unterstützt die folgenden Einrichtungen:
  - Ein AWS Directory Service Active Directory. AWS Directory Service ist ein AWS verwaltetes Active Directory, das auf Amazon EC2 gehostet wird. Weitere Informationen finden Sie unter Erste Schritte mit AWS Managed Microsoft AD im AWS Directory Service Administratorhandbuch.
  - Ein On-Premises-Active-Directory. Sie müssen sicherstellen, dass die Container-Instance von Amazon ECS Linux der Domain beitreten kann. Weitere Informationen finden Sie unter AWS Direct Connect.
- Sie haben ein bestehendes gMSA-Konto im Active Directory. Weitere Informationen finden Sie unter gMSAs für Windows-Container erstellen.
- Sie haben sich für domainlose gMSA entschieden, oder die Container-Instance von Amazon ECS Windows, die die Amazon-ECS-Aufgabe hostet, muss mit der Domain des Active Directory

verbunden sein und Mitglied der Active-Directory-Sicherheitsgruppe sein, die Zugriff auf das gMSA-Konto hat.

Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

- Sie haben die erforderlichen IAM-Berechtigungen hinzugefügt. Welche Berechtigungen erforderlich sind, hängt von den Methoden ab, die Sie für die anfänglichen Anmeldeinformationen und für die Speicherung der Anmeldeinformation wählen:
  - Wenn Sie Domainless gMSA für anfängliche Anmeldeinformationen verwenden, AWS Secrets Manager sind IAM-Berechtigungen für für die Amazon EC2 EC2-Instance-Rolle erforderlich.
  - Wenn Sie die Spezifikation der Anmeldeinformationen im SSM Parameter Store speichern, sind IAM-Berechtigungen für den Amazon EC2 Systems Manager Parameter Store für die Rolle der Aufgabenausführung erforderlich.
  - Wenn Sie die Spezifikation der Anmeldeinformationen in Amazon S3 speichern, sind IAM-Berechtigungen für Amazon Simple Storage Service für die Rolle der Aufgabenausführung erforderlich.

Einrichten von gMSA für Windows-Container auf Amazon ECS

Um gMSA für Windows-Container auf Amazon ECS einzurichten, können Sie dem vollständigen Tutorial folgen, das auch die Konfiguration der Voraussetzungen beinhaltet, Verwenden von Amazon ECS-Windows-Containern mit domainless unter gMSA Verwendung von AWS CLI.

In den folgenden Abschnitten wird die CredSpec-Konfiguration im Detail behandelt.

Themen

- Beispiel-CredSpec
- Domainlose gMSA einrichten
- Referenzieren einer Anmeldeinformations-Spezifikationsdatei in einer Aufgabendefinition

Beispiel-CredSpec

Amazon ECS verwendet eine Anmeldespezifikationsdatei, die die gMSA-Metadaten enthält, die verwendet werden, um den gMSA-Kontokontext an den Windows-Container zu übertragen. Sie

können die Anmeldeinformations-Spezifikationsdatei generieren und im Feld `credentialSpec` in Ihrer Aufgabendefinition darauf verweisen. Die Anmeldeinformations-Spezifikationsdatei enthält keine Secrets.

Im Folgenden finden Sie eine Beispiel-Anmeldeinformations-Spezifikationsdatei:

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "contoso.com",
        "DnsName": "contoso.com",
        "NetBiosName": "contoso"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "contoso.com"
            }
        ]
    }
}
```

Domainlose gMSA einrichten

Wir empfehlen domainlose gMSA, anstatt die Container-Instances zu einer einzigen Domain zu verbinden. Durch die Verwendung von domainlosen gMSA ist die Container-Instance nicht mit der Domain verbunden, andere Anwendungen auf der Instance können die Anmeldeinformationen nicht verwenden, um auf die Domain zuzugreifen, und Aufgaben, die verschiedenen Domains angehören, können auf derselben Instance ausgeführt werden.

1. Bevor Sie die CredSpec in eine der Speicheroptionen hochladen, fügen Sie Informationen mit dem ARN des Secrets im Secrets Manager oder SSM Parameter Store zur CredSpec hinzu. Weitere Informationen finden Sie unter Konfiguration zusätzlicher Anmeldeinformationen für den Anwendungsfall non-domain-joined Container-Hosts auf der Microsoft Learn-Website.

Format der Anmeldeinformationen für domainlose gMSA

Das Folgende ist das JSON-Format für die domainlosen gMSA-Anmeldeinformationen für Ihr Active Directory. Speichern Sie die Anmeldeinformationen in Secrets Manager oder SSM Parameter Store.

```
{
    "username":"WebApp01",
    "password":"Test123!",
    "domainName":"contoso.com"
}
```

2. Fügen Sie der CredSpec-Datei in der `ActiveDirectoryConfig` die folgende Information hinzu. Ersetzen Sie den ARN durch das Secret in Secrets Manager oder SSM Parameter Store.

   Beachten Sie, dass der `PluginGUID`-Wert mit der GUID im folgenden Beispielausschnitt übereinstimmen muss und erforderlich ist.

```
    "HostAccountConfig": {
        "PortableCcgVersion": "1",
        "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
        "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

   Sie können auch ein Secret im SSM Parameter Store verwenden, indem Sie den ARN in diesem Format verwenden: `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

3. Nachdem Sie die CredSpec-Datei geändert haben, sollte sie wie im folgenden Beispiel aussehen:

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "WebApp01",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
```

```
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      },
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Referenzieren einer Anmeldeinformations-Spezifikationsdatei in einer Aufgabendefinition

Amazon ECS unterstützt die folgenden Möglichkeiten, um auf den Dateipfad im `credentialSpecs`-Feld der Aufgabendefinition zu verweisen. Für jede dieser Optionen können Sie `credentialspec:` oder `domainlesscredentialspec:` angeben, je nachdem, ob Sie die Container-Instances zu einer einzigen Domain zusammenfügen oder domainlose gMSA verwenden.

Amazon S3 Bucket

Fügen Sie die Anmeldeinformationsspezifikation zu einem Amazon S3 Bucket hinzu und verweisen Sie dann im Feld `credentialSpecs` der Aufgabendefinition auf den Amazon-Ressourcennamen (ARN) des Amazon S3 Buckets.

```
{
    "family": "",
    "executionRoleArn": "",
```

```
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Sie müssen auch die folgenden Berechtigungen als Inline-Richtlinie zur IAM-Rolle der Amazon-ECS-Aufgabenausführung hinzufügen, um Ihren Aufgaben Zugriff auf den Amazon S3 Bucket zu geben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

SSM-Parameterspeicher-Parameter

Fügen Sie die Anmeldeinformationsspezifikation zu einem SSM-Parameterspeicher-Parameter hinzu und verweisen Sie dann im Feld `credentialSpecs` der Aufgabendefinition auf den Amazon-Ressourcennamen (ARN) des SSM-Parameterspeicher-Parameters.

```
{
    "family": "",
```

```
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [

 "credentialspecdomainless:arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Sie müssen außerdem die folgenden Berechtigungen als Inline-Richtlinie zur IAM-Rolle der Amazon-ECS-Aufgabenausführung hinzufügen, um Ihren Aufgaben Zugriff auf den SSM-Parameterspeicher-Parameter zu geben.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Lokale Datei

Wenn die Daten der Anmeldeinformationsspezifikation in einer lokalen Datei enthalten sind, verweisen Sie auf den Dateipfad im Feld `credentialSpecs` der Aufgabendefinition. Der Dateipfad, auf den verwiesen wird, muss relativ zum `C:\ProgramData\Docker\CredentialSpecs`-Verzeichnis sein und den umgekehrten Schrägstrich („\“) als Dateipfadtrennzeichen verwenden.

```
{
```

```
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspec:file://CredentialSpecDir\CredentialSpecFile.json"
            ],
            ...
        }
    ],
    ...
}
```

Verwenden von EC2 Image Builder zur Erstellung benutzerdefinierter Amazon ECS-optimierter AMIs

AWS empfiehlt die Verwendung der Amazon ECS-optimierten AMIs, da diese mit den Anforderungen und Empfehlungen für die Ausführung Ihrer Container-Workloads vorkonfiguriert sind. Es kann vorkommen, dass Sie Ihr AMI anpassen müssen, um zusätzliche Software hinzuzufügen. Sie können EC2 Image Builder für die Erstellung, Verwaltung und Bereitstellung Ihrer Server-Images verwenden. Sie behalten das Eigentum an den benutzerdefinierten Images, die in Ihrem Konto erstellt wurden. Sie können EC2 Image Builder Builder-Pipelines verwenden, um Updates und System-Patches für Ihre Images zu automatisieren, oder einen eigenständigen Befehl verwenden, um ein Image mit Ihren definierten Konfigurationsressourcen zu erstellen.

Sie erstellen ein Rezept für Ihr Image. Das Rezept enthält ein übergeordnetes Bild und alle zusätzlichen Komponenten. Sie erstellen auch eine Pipeline, die Ihr benutzerdefiniertes AMI verteilt.

Sie erstellen ein Rezept für Ihr Image. Ein Image Builder Builder-Image-Rezept ist ein Dokument, das das Basis-Image und die Komponenten definiert, die auf das Basis-Image angewendet werden, um die gewünschte Konfiguration für das Ausgabe-AMI-Image zu erzeugen. Sie erstellen auch eine Pipeline, die Ihr benutzerdefiniertes AMI verteilt. Weitere Informationen finden Sie unter So funktioniert EC2 Image Builder im EC2 Image Builder Builder-Benutzerhandbuch.

Wir empfehlen, eines der folgenden Amazon ECS-optimierten AMIs als Ihr „übergeordnetes Image“ in EC2 Image Builder zu verwenden:

- Linux

  - Amazon ECS-optimiertes AL2023 x86
  - Amazon-ECS-optimiertes AMI für Amazon Linux 2023 (arm64)
  - Amazon ECS-optimiertes Amazon Linux 2-Kernel-5-AMI
  - Amazon ECS-optimiertes Amazon Linux 2 x86-AMI
- Windows
  - Amazon ECS-optimiertes Windows 2022 Vollversion x86
  - Amazon ECS-optimiertes Windows 2022 Core x86
  - Amazon ECS-optimiertes Windows 2019 Full x86
  - Amazon ECS-optimiertes Windows 2019 Core x86
  - Amazon ECS-optimiertes Windows 2016 Full x86

Wir empfehlen außerdem, dass Sie „Die neueste verfügbare Betriebssystemversion verwenden“ auswählen. Die Pipeline verwendet für das übergeordnete Image eine semantische Versionierung, die hilft, die Abhängigkeitsupdates in automatisch geplanten Jobs zu erkennen. Weitere Informationen finden Sie unter Semantische Versionierung im EC2 Image Builder Builder-Benutzerhandbuch.

AWS aktualisiert regelmäßig Amazon ECS-optimierte AMI-Images mit Sicherheitspatches und der neuen Container-Agent-Version. Wenn Sie eine AMI-ID als Ihr übergeordnetes Image in Ihrem Image-Rezept verwenden, müssen Sie regelmäßig nach Aktualisierungen für das übergeordnete Image suchen. Wenn es Updates gibt, müssen Sie eine neue Version Ihres Rezepts mit dem aktualisierten AMI erstellen. Dadurch wird sichergestellt, dass Ihre benutzerdefinierten Images die neueste Version des übergeordneten Images enthalten. Informationen zum Erstellen eines Workflows zur automatischen Aktualisierung Ihrer EC2-Instances in Ihrem Amazon ECS-Cluster mit den neu erstellten AMIs finden Sie unter So erstellen Sie eine AMI-Hardening-Pipeline und automatisieren Updates für Ihre ECS-Instance-Flotte.

Sie können auch den Amazon-Ressourcennamen (ARN) eines übergeordneten Images angeben, das über eine verwaltete EC2 Image Builder Builder-Pipeline veröffentlicht wurde. Amazon veröffentlicht routinemäßig Amazon ECS-optimierte AMI-Images über verwaltete Pipelines. Diese Bilder sind öffentlich zugänglich. Sie müssen über die richtigen Berechtigungen verfügen, um auf das Bild zugreifen zu können. Wenn Sie in Ihrem Image Builder Builder-Rezept einen Image-ARN anstelle eines AMI verwenden, verwendet Ihre Pipeline bei jeder Ausführung automatisch die neueste Version des übergeordneten Images. Dieser Ansatz macht es überflüssig, für jedes Update manuell neue Rezeptversionen zu erstellen.

Verwenden des Image-ARN mit Infrastruktur als Code (IaC)

Sie können das Rezept mit der EC2 Image Builder Builder-Konsole, Infrastructure as Code (z. B. AWS CloudFormation) oder dem AWS SDK konfigurieren. Wenn Sie in Ihrem Rezept ein übergeordnetes Image angeben, können Sie eine EC2-AMI-ID, einen Image Builder Builder-Image-ARN, eine AWS Marketplace Produkt-ID oder ein Container-Image angeben. AWS veröffentlicht sowohl AMI-IDs als auch Image Builder Builder-Image-ARNs von Amazon ECS-optimierten AMIs öffentlich. Das folgende ist das ARN-Format für das Bild:

```
arn:${Partition}:imagebuilder:${Region}:${Account}:image/${ImageName}/${ImageVersion}
```

Das `ImageVersion` hat das folgende Format. Ersetzen Sie *`Major`*, *`Minor`* und *`Patch`* durch die neuesten Werte.

```
<major>.<minor>.<patch>
```

Sie können `minor` und `patch` durch bestimmte Werte ersetzen `major` oder Sie können den versionslosen ARN eines Images verwenden, sodass Ihre Pipeline up-to-date bei der neuesten Version des übergeordneten Images bleibt. Ein versionsloser ARN verwendet das Platzhalterformat 'x.x.x', um die Image-Version darzustellen. Dieser Ansatz ermöglicht es dem Image Builder Builder-Dienst, automatisch die neueste Version des Images aufzulösen. Durch die Verwendung von versionsfreiem ARN wird sichergestellt, dass Ihre Referenz immer auf das neueste verfügbare Image verweist, wodurch der Prozess der Aktualisierung von Images in Ihrer Bereitstellung optimiert wird. Wenn Sie mit der Konsole ein Rezept erstellen, identifiziert EC2 Image Builder automatisch den ARN für Ihr übergeordnetes Image. Wenn Sie IaC verwenden, um das Rezept zu erstellen, müssen Sie den ARN identifizieren und die gewünschte Image-Version auswählen oder den versionslosen ARN verwenden, um das neueste verfügbare Image anzugeben. Wir empfehlen Ihnen, ein automatisiertes Skript zu erstellen, um nur Bilder zu filtern und anzuzeigen, die Ihren Kriterien entsprechen. Das folgende Python-Skript zeigt, wie Sie eine Liste von Amazon ECS-optimierten AMIs abrufen.

Das Skript akzeptiert zwei optionale Argumente: `owner` und`platform`, mit den Standardwerten „Amazon“ bzw. „Windows“. Die gültigen Werte für das Argument owner sind: `SelfShared`,`Amazon`, und`ThirdParty`. Die gültigen Werte für das Plattformargument sind `Windows` und`Linux`. Wenn Sie das Skript beispielsweise mit den `owner` Argumenten set to `Amazon` und set to ausführen, generiert das `platform` Skript eine Liste von Bildern`Linux`, die von Amazon veröffentlicht wurden, einschließlich Amazon ECS-optimierter Bilder.

```
import boto3
```

```
import argparse

def list_images(owner, platform):
    # Create a Boto3 session
    session = boto3.Session()

    # Create an EC2 Image Builder client
    client = session.client('imagebuilder')

    # Define the initial request parameters
    request_params = {
        'owner': owner,
        'filters': [
            {
                'name': 'platform',
                'values': [platform]
            }
        ]
    }

    # Initialize the results list
    all_images = []

    # Get the initial response with the first page of results
    response = client.list_images(**request_params)

    # Extract images from the response
    all_images.extend(response['imageVersionList'])

    # While 'nextToken' is present, continue paginating
    while 'nextToken' in response and response['nextToken']:
        # Update the token for the next request
        request_params['nextToken'] = response['nextToken']

        # Get the next page of results
        response = client.list_images(**request_params)

        # Extract images from the response
        all_images.extend(response['imageVersionList'])

    return all_images

def main():
    # Initialize the parser
```

```
    parser = argparse.ArgumentParser(description="List AWS images based on owner and
 platform")

    # Add the parameters/arguments
    parser.add_argument("--owner", default="Amazon", help="The owner of the images.
 Default is 'Amazon'.")
    parser.add_argument("--platform", default="Windows", help="The platform type of the
 images. Default is 'Windows'.")

    # Parse the arguments
    args = parser.parse_args()

    # Retrieve all images based on the provided owner and platform
    images = list_images(args.owner, args.platform)

    # Print the details of the images
    for image in images:
        print(f"Name: {image['name']}, Version: {image['version']}, ARN:
 {image['arn']}")

if __name__ == "__main__":
    main()
```

Verwenden des Bild-ARN mit AWS CloudFormation

Ein Image Builder Builder-Image-Rezept ist ein Blueprint, der das übergeordnete Image und die Komponenten spezifiziert, die erforderlich sind, um die beabsichtigte Konfiguration des Ausgabe-Images zu erreichen. Sie verwenden die `AWS::ImageBuilder::ImageRecipe` Ressource. Stellen Sie den `ParentImage` Wert auf den Bild-ARN ein. Verwenden Sie den versionslosen ARN Ihres gewünschten Images, um sicherzustellen, dass Ihre Pipeline immer die neueste Version des Images verwendet. z. B. `arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-optimized-x86/x.x.x`. Die folgende `AWS::ImageBuilder::ImageRecipe` Ressourcendefinition verwendet einen von Amazon verwalteten Image ARN:

```
ECSRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
    Properties:
      Name: MyRecipe
      Version: '1.0.0'
      Components:
        - ComponentArn: [<The component arns of the image recipe>]
```

```
      ParentImage: "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-
optimized-x86/x.x.x"
```

Weitere Informationen zu der AWS::ImageBuilder::ImageRecipeRessource finden Sie im AWS CloudFormation Benutzerhandbuch.

Sie können die Erstellung neuer Bilder in Ihrer Pipeline automatisieren, indem Sie die `Schedule` Eigenschaft der `AWS::ImageBuilder::ImagePipeline` Ressource festlegen. Der Zeitplan enthält eine Startbedingung und einen Cron-Ausdruck. Weitere Informationen finden Sie unter AWS::ImageBuilder::ImagePipeline im AWS CloudFormation -Benutzerhandbuch.

Im folgenden `AWS::ImageBuilder::ImagePipeline` Beispiel wird für die Pipeline täglich um 10:00 Uhr Coordinated Universal Time (UTC) ein Build ausgeführt. Legen Sie die folgenden `Schedule` Werte fest:

- Setzen Sie `PipelineExecutionStartCondition` auf `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. Der Build wird nur initiiert, wenn abhängige Ressourcen wie das übergeordnete Image oder Komponenten, die in ihren semantischen Versionen den Platzhalter „x“ verwenden, aktualisiert werden. Dadurch wird sichergestellt, dass der Build die neuesten Updates dieser Ressourcen enthält.
- Auf ScheduleExpression den Cron-Ausdruck `(0 10 * * ? *)` gesetzt.

```
ECSPipeline:
    Type: AWS::ImageBuilder::ImagePipeline
    Properties:
      Name: my-pipeline
      ImageRecipeArn: <arn of the recipe you created in previous step>
      InfrastructureConfigurationArn: <ARN of the infrastructure configuration
 associated with this image pipeline>
      Schedule:
        PipelineExecutionStartCondition:
 EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE
        ScheduleExpression: 'cron(0 10 * * ? *)'
```

Verwenden des Bild-ARN mit Terraform

Der Ansatz zur Angabe des übergeordneten Images und des Zeitplans Ihrer Pipeline in Terraform entspricht dem in. AWS CloudFormation Sie verwenden die Ressource.

`aws_imagebuilder_image_recipe` Stellen Sie den `parent_image` Wert auf den Bild-ARN ein. Verwenden Sie den versionslosen ARN Ihres gewünschten Images, um sicherzustellen, dass Ihre Pipeline immer die neueste Version des Images verwendet. Weitere Informationen finden Sie `aws_imagebuilder_image_recipe`in der Terraform-Dokumentation.

Setzen Sie im Zeitplankonfigurationsblock von den den `schedule_expression` Argumentwert auf einen Cron-Ausdruck Ihrer Wahl`aws_imagebuilder_image_pipeline` `resource`, um anzugeben, wie oft die Pipeline ausgeführt wird, und setzen Sie den Wert auf. `pipeline_execution_start_condition` `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE` Weitere Informationen finden Sie `aws_imagebuilder_image_pipeline`in der Terraform-Dokumentation.

Verwenden von AWS Deep Learning Containers auf Amazon ECS

AWS Deep Learning Containers bieten eine Reihe von Docker-Images für das Training und die Bereitstellung von Modellen in TensorFlow und Apache MXNet (Incubating) auf Amazon ECS. Deep Learning Containers ermöglichen optimierte Umgebungen mit TensorFlow NVIDIA CUDA- (für GPU-Instances) und Intel MKL- (für CPU-Instances) Bibliotheken. Containerbilder für Deep Learning Containers sind in Amazon ECR verfügbar, um in Amazon ECS-Aufgabendefinitionen zu referenzieren. Sie können Deep Learning Containers zusammen mit Amazon Elastic Inference verwenden, um Ihre Inferenzkosten zu senken.

Informationen zu den ersten Schritten mit Deep Learning Containers ohne Elastic Inference in Amazon ECS finden Sie unter Deep Learning Container in Amazon ECS im AWS Deep Learning AMI -Entwicklerhandbuch.

Deep Learning Containers mit Elastic Inference auf Amazon ECS

> **Note**
>
> Ab 15. April 2023 AWS wird Amazon Elastic Inference (EI) keine Neukunden mehr in Amazon Elastic Inference (EI) einbinden und Bestandskunden helfen, ihre Workloads auf Optionen umzustellen, die ein besseres Preis und eine bessere Leistung bieten. Nach dem 15. April 2023 können Neukunden keine Instances mit Amazon EI-Beschleunigern in Amazon SageMaker, Amazon ECS oder Amazon EC2 starten. Kunden, die Amazon EI in den letzten 30 Tagen mindestens einmal genutzt haben, gelten jedoch als aktuelle Kunden und können den Service weiterhin nutzen.

AWS Deep Learning Containers bieten eine Reihe von Docker-Images für die Bereitstellung von Modellen in TensorFlow und Apache MXNet (Incubating), die die Amazon Elastic Inference Inference-Beschleuniger nutzen. Amazon ECS bietet Task-Definitionsparameter zum Anhängen von Elastic Inference-Accelerators an Ihre Container. Wenn Sie einen Elastic Inference-Accelerator-Typ in Ihrer Aufgabendefinition angeben, verwaltet Amazon ECS den Lebenszyklus und die Konfiguration des Accelerators. Die service-verknüpfte Amazon ECS-Rolle ist erforderlich, wenn Sie dieses Feature verwenden. Weitere Informationen zu Elastic Inference-Accelerators finden Sie unter Amazon Elastic Inference – Grundlagen.

> **Important**
>
> Für Ihre Amazon ECS-Container-Instances ist mindestens Version `1.30.0` des Container-Agenten erforderlich. Informationen zum Überprüfen Ihrer Agenten-Version und zum Aktualisieren auf die neueste Version finden Sie unter Überprüfen des Amazon-ECS-Container-Agenten.

Informationen zu den ersten Schritten mit Elastic Inference in Amazon ECS-Deep Learning Containers in finden Sie unter Deep Learning Containers mit Elastic Inference auf Amazon ECS im Amazon Elastic Inference-Entwicklerhandbuch.

Amazon-ECS-Service-Kontingente

Die folgende Tabelle enthält die Standard-Service Quotas, die auch als Limits bezeichnet werden, für Amazon ECS für ein AWS-Konto-Konto. Weitere Informationen über die Servicekontingente für andere AWS-Services , die Sie mit Amazon ECS nutzen können, wie Elastic Load Balancing und Auto Scaling, finden Sie unter AWS Service Quotas im Allgemeine Amazon Web Services-Referenz. Weitere Informationen zur API-Drosselung im Amazon-ECS-API finden Sie unter Anforderung der Drosselung für die Amazon-ECS-API.

Amazon-ECS-Service-Kontingente

Im Folgenden finden Sie Amazon-ECS-Servicekontingente.

Neue AWS Konten haben möglicherweise anfänglich niedrigere Kontingente, die sich im Laufe der Zeit erhöhen können. Amazon ECS überwacht ständig die Kontonutzung in jeder Region und erhöht dann automatisch die Kontingente basierend auf Ihrer Nutzung. Sie können auch eine Kontingenterhöhung für Werte beantragen, die als anpassbar angezeigt werden, siehe Beantragen einer Kontingenterhöhung im Service Quotas — Benutzerhandbuch.

Name	Standard	Anpas	Beschreibung
Kapazitätsanbieter pro Cluster	Jede unterstützte Region: 20	Nein	Die maximale Anzahl von Kapazitätsanbietern, die einem Cluster zugeordnet werden können.
Classic Load Balancer pro Service	Jede unterstützte Region: 1	Nein	Die maximale Anzahl von Classic Load Balancern pro Service.
Clusters per account (Cluster pro Konto)	Jede unterstützte Region: 10 000	Ja	Anzahl Cluster pro Konto
Container-Instances pro Cluster	Jede unterstützte Region: 5 000	Nein	Anzahl Container-Instances pro Cluster
Container-Instances pro Start-Aufgabe	Jede unterstützte Region: 10	Nein	Die maximale Anzahl von Container-Instances, die

Name	Standard	Anpas	Beschreibung
			in einer StartTask API-Aktion angegeben ist.
Container pro Aufgabendefinition	Jede unterstützte Region: 10	Nein	Die maximale Anzahl der Containerdefinitionen innerhalb einer Aufgabendefinition.
ECS Exec-Sitzungen	Jede unterstützte Region: 1 000	Nein	Die maximale Anzahl von ECS Exec-Sitzungen pro Container.
Rate der Aufgaben, die von einem Dienst auf AWS Fargate gestartet wurden	Jede unterstützte Region: 500	Nein	Die maximale Anzahl von Aufgaben, die pro Service pro Minute auf Fargate vom Amazon-ECS-Service-Scheduler bereitgestellt werden können.
Die Rate der Aufgaben, die von einem Service auf einer Amazon-EC2-Instance oder einer externen Instance gestartet wurden	Jede unterstützte Region: 500	Nein	Die maximale Anzahl von Aufgaben, die vom Amazon-ECS-Service-Scheduler pro Service und Minute auf einer Amazon-EC2-Instance oder externen Instance bereitgestellt werden können.

Name	Standard	Anpas	Beschreibung
Revisionen pro Aufgabendefinitionsfamilie	Jede unterstützte Region: 1 000 000	Nein	Die maximale Anzahl der Revisionen pro Aufgabendefinitionsfamilie. Das Abmelden oder Löschen einer Revision einer Aufgabendefinition schließt nicht aus, dass sie in dieses Limit aufgenommen wird.
Sicherheitsgruppen pro awsvpcConfiguration	Jede unterstützte Region: 5	Nein	Die maximale Anzahl von Sicherheitsgruppen, die innerhalb einer awsvpcConfiguration angegeben werden.
Services per cluster (Services pro Cluster)	Jede unterstützte Region: 5 000	Nein	Die maximale Anzahl von Services pro Cluster
Services pro Namespace	Jede unterstützte Region: 100	Yes (Ja)	Die maximale Anzahl von Services, die in einem Namespace ausgeführt werden können.
Subnetze pro awsvpcConfiguration	Jede unterstützte Region: 16	Nein	Die maximale Anzahl von Subnetzen, die innerhalb einer awsvpcConfiguration angegeben werden.
Tags pro Ressource	Jede unterstützte Region: 50	Nein	Die maximale Anzahl an Tags pro Ressource. Dies gilt für Aufgabendefinitionen, Cluster, Aufgaben und Services.

Name	Standard	Anpas	Beschreibung
Zielgruppen pro Service	Jede unterstützte Region: 5	Nein	Die maximale Anzahl von Zielgruppen pro Service, wenn ein Application Load Balancer oder ein Network Load Balancer verwendet wird.
Größe der Aufgabendefinition	Jede unterstützte Region: 64 Kilobyte	Nein	Die maximale Größe einer Aufgabendefinition in KiB.
Aufgaben im Zustand PROVISIONING pro Cluster	Jede unterstützte Region: 500	Nein	Die maximale Anzahl von Aufgaben, die pro Cluster im PROVISIONING-Zustand warten. Dieses Kontingent gilt nur für Aufgaben, die mit einem Kapazitätsanbieter der EC2-Auto-Scaling-Gruppe gestartet wurden.
Pro Ausführungsaufgabe gestartete Aufgaben	Jede unterstützte Region: 10	Nein	Die maximale Anzahl von Aufgaben, die pro RunTask API-Aktion gestartet werden können.
Aufgaben pro Service	Jede unterstützte Region: 5 000	Nein	Die maximale Anzahl der Aufgaben pro Services (die gewünschte Anzahl).

Note

Bei den Standardwerten handelt es sich um die anfänglichen Kontingente AWS, die vom tatsächlich angewendeten Kontingentwert und dem maximal möglichen Servicekontingent

getrennt sind. Weitere Informationen finden Sie unter Terminologie in Service Quotas im Service Quotas User Guide.

> **Note**
>
> Services, die für die Verwendung von Amazon ECS Service Discovery konfiguriert sind, haben ein Limit von 1 000 Aufgaben pro Service. Der Grund hierfür ist das AWS Cloud Map - Servicekontingent für die Anzahl der Instances pro Service. Weitere Informationen finden Sie unter AWS Cloud Map Servicekontingente im Allgemeine Amazon Web Services-Referenz.

> **Note**
>
> In der Praxis hängen die Launch-Raten von Aufgaben auch von anderen Überlegungen ab, wie zum Beispiel herunterzuladende und zu entpackende Container-Images, Zustandsprüfungen und andere aktivierte Integrationen, wie das Registrieren von Aufgaben in einem Load Balancer. Möglicherweise werden Sie Schwankungen in den Launch-Raten der Aufgaben im Vergleich zu den hier dargestellten Kontingenten sehen. Diese Abweichungen werden durch die Funktionen verursacht, die Sie für Ihre Dienste verwenden. Weitere Informationen finden Sie unter Bewährte Methoden für Amazon ECS-Serviceparameter .

> **Note**
>
> Services, die für die Verwendung von Amazon ECS Service Connect konfiguriert sind, haben ein Limit von 1 000 Aufgaben pro Service. Dies ist auf das AWS Cloud Map Dienstkontingent für die Anzahl der Instanzen pro Dienst zurückzuführen. Weitere Informationen finden Sie unter AWS Cloud Map Servicekontingente im Allgemeine Amazon Web Services-Referenz.

AWS Fargate Servicekontingenten

Im Folgenden finden Sie Amazon ECS auf AWS Fargate Service Quotas, die unter dem AWS FargateService in der Service Quotas-Konsole aufgeführt sind.

Für neue AWS Konten gelten möglicherweise zunächst niedrigere Kontingente, die sich im Laufe der Zeit erhöhen können. Fargate überwacht ständig die Kontonutzung in jeder Region und erhöht dann automatisch die Kontingente basierend auf Ihrer Nutzung. Sie können auch eine Kontingenterhöhung für Werte beantragen, die als anpassbar angezeigt werden, siehe Beantragen einer Kontingenterhöhung im Service Quotas — Benutzerhandbuch.

Name	Standard	Anpas	Beschreibung
Anzahl der On-Demand-vCPU-Ressourcen von Fargate	Jede unterstützte Region: 6	Ja	Die Anzahl der Fargate-vCPUs, die in diesem Konto in der aktuellen Region gleichzeitig als Fargate On-Demand ausgeführt werden.
Anzahl der vCPU-Spot-Ressourcen von Fargate	Jede unterstützte Region: 6	Ja	Die Anzahl der Fargate-vCPUs, die in diesem Konto in der aktuellen Region gleichzeitig als Fargate-Spot-Ressourcen ausgeführt werden.

Note

Bei den Standardwerten handelt es sich um die anfänglichen Kontingente AWS, die vom tatsächlich angewendeten Kontingent und dem maximal möglichen Servicekontingent getrennt sind. Weitere Informationen finden Sie unter Terminologie in Service Quotas im Service Quotas User Guide.

Note

Fargate erzwingt zusätzlich Launch-Ratenlimits für Amazon-ECS-Aufgaben und Amazon-EKS-Pods. Weitere Informationen finden Sie unter Drosselungs-Limits bei Fargate.

Verwaltung Ihrer Amazon ECS- und AWS Fargate Servicekontingenten in der AWS Management Console

Amazon ECS ist in Service Quotas integriert, einen AWS Service, mit dem Sie Ihre Kontingente von einem zentralen Ort aus einsehen und verwalten können. Weitere Informationen zu Service Quotas finden Sie unter Was sind Service Quotas im Benutzerhandbuch für Service Quotas.

Mit Service Quotas können Sie den Wert Ihrer Amazon-ECS-Servicekontingente einfach ermitteln.

AWS Management Console

So zeigen Sie Amazon-ECS- und Fargate-Servicekontingente mit AWS Management Console an

1. Öffnen Sie die Service-Quotas-Konsole unter https://console.aws.amazon.com/servicequotas/.
2. Wählen Sie im Navigationsbereich AWS -Services.
3. Suchen Sie auf der AWS -Services-Liste nach Amazon Elastic Container Service (Amazon ECS) oder AWS Fargate.

   In der Liste Service Quotas wird der Name der Service Quota, der angewendete Wert (falls verfügbar) und das AWS -Standardkontingent angezeigt. Zudem wird angezeigt, ob der Kontingentwert anpassbar ist.
4. Wählen Sie den Kontingentnamen, um zusätzliche Informationen zu einem Service Quota anzuzeigen, z. B. seine Beschreibung.
5. (Optional) Um eine Kontingenterhöhung zu beantragen, wählen Sie das Kontingent, das Sie erhöhen möchten, und dann Request quota increase (Kontingenterhöhung beantragen) aus, geben Sie die erforderlichen Informationen ein, und wählen Sie dann Request (Beantragen) aus.

Weitere Informationen zum Umgang mit Servicekontingenten AWS Management Console finden Sie im Service Quotas User Guide. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter Anfordern einer Kontingenterhöhung im Bnutzerhandbuch zu Service Quotas.

AWS CLI

So zeigen Sie Amazon-ECS- und Fargate-Servicekontingente mit AWS CLI an

Führen Sie den folgenden Befehl aus, um die standardmäßigen Amazon-ECS-Quotas anzuzeigen.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code ecs \
    --output table
```

Führen Sie den folgenden Befehl aus, um die standardmäßigen Fargate-Kontingente anzuzeigen.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

Führen Sie den folgenden Befehl aus, um Ihre angewendeten Fargate-Kontingente anzuzeigen.

```
aws service-quotas list-service-quotas \
    --service-code fargate
```

> **Note**
>
> Amazon ECS unterstützt keine angewendeten Kontingente.

Weitere Informationen zum Arbeiten mit Service Quotas mithilfe von finden Sie in der AWS CLIAWS CLI Befehlsreferenz für Dienstkontingente. Informationen zum Beantragen einer Kontingenterhöhung finden Sie unter dem `request-service-quota-increase`-Befehl in der AWS CLI -Befehlsreferenz.

Verwaltung von Amazon ECS-Servicekontingenten und API-Drosselungslimits

Amazon ECS ist in mehrere Systeme integriert AWS-Services, darunter Elastic Load Balancing und Amazon EC2. AWS Cloud Map Durch diese enge Integration umfasst Amazon ECS mehrere Funktionen wie Service Load Balancing, Service Connect, Task Networking und auto

Clusterskalierung. Amazon ECS und das andere AWS-Services , das es in alle integriert, verwalten Service-Kontingente und API-Ratenbegrenzungen, um eine konsistente Leistung und Auslastung zu gewährleisten. Diese Servicekontingente verhindern auch, dass versehentlich mehr Ressourcen als benötigt bereitgestellt werden, und schützen vor böswilligen Aktionen, die Ihre Rechnung erhöhen könnten.

Wenn Sie sich mit Ihren Servicekontingenten und den AWS API-Ratenbegrenzungen vertraut machen, können Sie die Skalierung Ihrer Workloads planen, ohne sich Gedanken über unerwartete Leistungseinbußen machen zu müssen. Weitere Informationen finden Sie unter Anforderungsdrosselung für die Amazon ECS-API.

Wir empfehlen Ihnen, bei der Skalierung Ihrer Workloads auf Amazon ECS die folgende Servicequote zu berücksichtigen.

- AWS Fargate verfügt über Kontingente, die jeweils die Anzahl der gleichzeitig ausgeführten Aufgaben begrenzen. AWS-Region Es gibt Kontingente sowohl für On-Demand-Aufgaben als auch für Fargate Spot-Aufgaben auf Amazon ECS. Jedes Servicekontingent umfasst auch alle Amazon EKS-Pods, die Sie auf Fargate ausführen.
- Für Aufgaben, die auf Amazon EC2 EC2-Instances ausgeführt werden, beträgt die maximale Anzahl von Amazon EC2 EC2-Instances, die Sie für jeden Cluster registrieren können, 5.000. Wenn Sie Amazon ECS-Cluster Auto Scaling mit einem Auto Scaling-Gruppenkapazitätsanbieter verwenden oder Amazon EC2 EC2-Instances für Ihren Cluster selbst verwalten, kann dieses Kontingent zu einem Engpass bei der Bereitstellung werden. Wenn Sie mehr Kapazität benötigen, können Sie mehr Cluster erstellen oder eine Erhöhung der Servicequote beantragen.
- Wenn Sie Amazon ECS-Cluster Auto Scaling mit einem Auto Scaling-Gruppenkapazitätsanbieter verwenden, berücksichtigen Sie bei der Skalierung Ihrer Services das `Tasks in the PROVISIONING state per cluster` Kontingent. Dieses Kontingent ist die maximale Anzahl von Aufgaben im `PROVISIONING` Bundesstaat für jeden Cluster, für die Kapazitätsanbieter die Kapazität erhöhen können. Wenn Sie eine große Anzahl von Aufgaben gleichzeitig starten, können Sie dieses Kontingent problemlos einhalten. Ein Beispiel ist die gleichzeitige Bereitstellung von Dutzenden von Diensten mit jeweils Hunderten von Aufgaben. In diesem Fall muss der Kapazitätsanbieter neue Container-Instances starten, um die Aufgaben zu platzieren, wenn der Cluster nicht genügend Kapazität hat. Während der Kapazitätsanbieter weitere Amazon EC2 EC2-Instances auf den Markt bringt, wird der Amazon ECS-Service Scheduler wahrscheinlich weiterhin Aufgaben parallel starten. Diese Aktivität kann jedoch aufgrund unzureichender Clusterkapazität gedrosselt werden. Der Amazon ECS Service Scheduler implementiert eine Back-off- und exponentielle Drosselungsstrategie, mit der versucht wird, Aufgaben erneut zu platzieren,

wenn neue Container-Instances gestartet werden. Infolgedessen kann es zu langsameren Bereitstellungs- oder Skalierungszeiten kommen. Um diese Situation zu vermeiden, können Sie Ihre Servicebereitstellungen in einer der folgenden Arten planen. Stellen Sie entweder eine große Anzahl von Aufgaben bereit, ohne dass die Clusterkapazität erhöht werden muss, oder Sie behalten freie Clusterkapazität für den Start neuer Aufgaben.

Berücksichtigen Sie bei der Skalierung Ihrer Workloads nicht nur das Amazon ECS-Servicekontingent, sondern auch das Servicekontingent für die anderen AWS-Services , die in Amazon ECS integriert sind.

Elastic Load Balancing

Sie können Ihre Amazon ECS-Services so konfigurieren, dass Elastic Load Balancing verwendet wird, um den Traffic gleichmäßig auf die Aufgaben zu verteilen. Weitere Informationen und empfohlene Best Practices für die Auswahl eines Load Balancers finden Sie unterVerwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.

Elastic Load Balancing Balancing-Dienstkontingente

Wenn Sie Ihre Workloads skalieren, sollten Sie die folgenden Elastic Load Balancing Service-Kontingente berücksichtigen. Die meisten Elastic Load Balancing Balancing-Dienstkontingente sind anpassbar, und Sie können eine Erhöhung in der Service Quotas-Konsole beantragen.

Application Load Balancer

Wenn Sie einen Application Load Balancer verwenden, müssen Sie je nach Anwendungsfall möglicherweise eine Erhöhung des Kontingents beantragen für:

- Das `Targets per Application Load Balancer` Kontingent, das die Anzahl der Ziele hinter Ihrem Application Load Balancer darstellt.
- Die `Targets per Target Group per Region` Quote, die der Anzahl der Ziele hinter Ihren Zielgruppen entspricht.

Weitere Informationen finden Sie unter Kontingente für Ihre Application Load Balancer im Benutzerhandbuch für Application Load Balancers.

Network Load Balancer

Die Anzahl der Ziele, die Sie bei einem Network Load Balancer registrieren können, ist strenger begrenzt. Wenn Sie einen Network Load Balancer verwenden, möchten Sie häufig die zonenübergreifende Unterstützung aktivieren, was zusätzliche Skalierungsbeschränkungen für `Targets per Availability Zone Per Network Load Balancer` die maximale Anzahl von Zielen pro Availability Zone für jeden Network Load Balancer mit sich bringt. Weitere Informationen finden Sie unter Kontingente für Ihre Network Load Balancer im Benutzerhandbuch für Network Load Balancer.

Drosselung der Elastic Load Balancing Balancing-API

Wenn Sie einen Amazon ECS-Service für die Verwendung eines Load Balancers konfigurieren, müssen die Zustandsprüfungen der Zielgruppe bestanden werden, bevor der Service als fehlerfrei eingestuft wird. Für die Durchführung dieser Zustandsprüfungen ruft Amazon ECS in Ihrem Namen Elastic Load Balancing API-Operationen auf. Wenn Sie in Ihrem Konto eine große Anzahl von Diensten mit Load Balancern konfiguriert haben, verlangsamen Sie möglicherweise die Servicebereitstellung, da es zu einer möglichen Drosselung speziell für die API-Operationen`RegisterTarget`,`DeregisterTarget`, und `DescribeTargetHealth` Elastic Load Balancing kommen kann. Wenn eine Drosselung auftritt, treten Drosselungsfehler in Ihren Amazon ECS-Serviceereignismeldungen auf.

Wenn Sie eine AWS Cloud Map API-Drosselung feststellen, können Sie sich an uns wenden AWS Support , um Hilfe zu erhalten, wie Sie Ihre API-Drosselungsgrenzen erhöhen können. AWS Cloud Map Weitere Informationen zur Überwachung und Behebung solcher Drosselungsfehler finden Sie unter. Probleme mit der Drosselung von Amazon ECS lösen

Elastic-Network-Schnittstelle

Wenn Ihre Aufgaben den `awsvpc` Netzwerkmodus verwenden, stellt Amazon ECS für jede Aufgabe eine eigene elastic network interface (ENI) bereit. Wenn Ihre Amazon ECS-Services einen Elastic Load Balancing Load Balancer verwenden, werden diese Netzwerkschnittstellen auch als Ziele für die entsprechende, im Service definierte Zielgruppe registriert.

Service-Kontingente für elastische Netzwerkschnittstellen

Wenn Sie Aufgaben ausführen, die den `awsvpc` Netzwerkmodus verwenden, wird jeder Aufgabe eine eindeutige elastic network interface zugewiesen. Wenn diese Aufgaben über das Internet erreicht werden müssen, weisen Sie der elastic network interface für diese Aufgaben eine öffentliche IP-Adresse zu. Wenn Sie Ihre Amazon ECS-Workloads skalieren, sollten Sie diese beiden wichtigen Kontingente berücksichtigen:

- Das `Network interfaces per Region` Kontingent, das die maximale Anzahl von Netzwerkschnittstellen in einem AWS-Region für Ihr Konto ist.
- Das `Elastic IP addresses per Region` Kontingent, das die maximale Anzahl elastischer IP-Adressen in einem darstellt AWS-Region.

Beide Service Quotas sind anpassbar, und Sie können für diese über Ihre Servicekontingenta-Konsole eine Erhöhung beantragen. Weitere Informationen finden Sie unter Amazon VPC Service Quotas im Amazon Virtual Private Cloud-Benutzerhandbuch.

Bei Amazon ECS-Workloads, die auf Amazon EC2 EC2-Instances gehostet werden, sollten Sie bei der Ausführung von Aufgaben, die den `awsvpc` Netzwerkmodus verwenden, das `Maximum network interfaces` Service-Kontingent, die maximale Anzahl von Netzwerk-Instances für jede Amazon EC2 EC2-Instance, berücksichtigen. Dieses Kontingent begrenzt die Anzahl der Aufgaben, die Sie auf einer Instance platzieren können. Sie können das Kontingent nicht anpassen und es ist in der Service Quotas Quotas-Konsole nicht verfügbar. Weitere Informationen finden Sie unter IP-Adressen pro Netzwerkschnittstelle pro Instance-Typ im Amazon EC2 EC2-Benutzerhandbuch.

Sie können zwar die Anzahl der Netzwerkschnittstellen, die an eine Amazon EC2 EC2-Instance angeschlossen werden können, nicht ändern, aber Sie können die elastic network interface Trunking-Funktion verwenden, um die Anzahl der verfügbaren Netzwerkschnittstellen zu erhöhen. Beispielsweise kann eine `c5.large` Instance standardmäßig bis zu drei Netzwerkschnittstellen haben. Die primäre Netzwerkschnittstelle für die Instance zählt dazu. Sie können also zwei weitere Netzwerkschnittstellen an die Instanz anhängen. Da für jede Aufgabe, die den `awsvpc` Netzwerkmodus verwendet, eine Netzwerkschnittstelle erforderlich ist, können Sie für diesen Instance-Typ in der Regel nur zwei solcher Aufgaben ausführen. Dies kann zu einer Unterauslastung Ihrer Clusterkapazität führen. Wenn Sie elastic network interface Trunking aktivieren, können Sie die Netzwerkschnittstellendichte erhöhen, um eine größere Anzahl von Aufgaben auf jeder Instance zu platzieren. Wenn Trunking aktiviert ist, kann eine `c5.large` Instance bis zu 12 Netzwerkschnittstellen haben. Die Instance hat die primäre Netzwerkschnittstelle und Amazon ECS erstellt und fügt der Instance eine „Trunk“ -Netzwerkschnittstelle hinzu. Daher können Sie mit dieser Konfiguration 10 Aufgaben auf der Instance ausführen, anstatt die standardmäßigen zwei Aufgaben. Weitere Informationen finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.

API-Drosselung der elastischen Netzwerkschnittstelle

Wenn Sie Aufgaben ausführen, die den `awsvpc` Netzwerkmodus verwenden, stützt sich Amazon ECS auf die folgenden Amazon EC2 EC2-APIs. Jede dieser APIs hat unterschiedliche API-

Drosselungen. Weitere Informationen finden Sie unter Anforderungsdrosselung für die Amazon EC2 EC2-API in der Amazon EC2 EC2-API-Referenz.

- CreateNetworkSchnittstelle
- AttachNetworkSchnittstelle
- DetachNetworkSchnittstelle
- DeleteNetworkSchnittstelle
- DescribeNetworkSchnittstellen
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGruppen
- DescribeInstances

Wenn die Amazon EC2 EC2-API-Aufrufe während der Workflows zur Bereitstellung von elastic network interface gedrosselt werden, versucht der Amazon ECS-Service Scheduler es automatisch mit exponentiellen Back-offs erneut. Diese automatischen Stilllegungen können manchmal zu Verzögerungen beim Starten von Aufgaben führen, was wiederum zu langsameren Bereitstellungsgeschwindigkeiten führt. Wenn es zu einer API-Drosselung kommt, wird die entsprechende Meldung in Ihren Service-Event-Meldungen `Operations are being throttled. Will try again later.` angezeigt. Wenn Sie die Amazon EC2 EC2-API-Drosselungen regelmäßig einhalten, können Sie sich an uns wenden AWS Support , um Unterstützung zu erhalten, wie Sie Ihre API-Drosselungsgrenzwerte erhöhen können. Weitere Informationen zur Überwachung und Behebung von Drosselungsfehlern finden Sie unter Umgang mit Drosselungsproblemen.

AWS Cloud Map

Amazon ECS Service Discovery und Service Connect verwenden AWS Cloud Map APIs, um Namespaces für Ihre Amazon ECS-Services zu verwalten. Wenn Ihre Services eine große Anzahl von Aufgaben haben, sollten Sie die folgenden Empfehlungen berücksichtigen.

AWS Cloud Map Dienstkontingente

Wenn Amazon ECS-Services für die Verwendung von Service Discovery oder Service Connect konfiguriert sind, wird das `Tasks per service` Kontingent, das die maximale Anzahl von Aufgaben für den Service darstellt, von dem AWS Cloud Map `Instances per service` Service-Kontingent beeinflusst, das die maximale Anzahl von Instances für diesen Service darstellt. Insbesondere

reduziert das AWS Cloud Map Service-Kontingent die Anzahl der Aufgaben, die Sie ausführen können, auf maximal 1.000 Serviceaufgaben. Sie können das AWS Cloud Map Kontingent nicht ändern. Weitere Informationen finden Sie unter AWS Cloud Map -Servicekontingente.

AWS Cloud Map API-Drosselung

Amazon ECS ruft die `DeregisterInstance` AWS Cloud Map APIs `ListInstancesGetInstancesHealthStatus`,`RegisterInstance`, und in Ihrem Namen auf. Sie helfen bei der Serviceerkennung und führen Integritätsprüfungen durch, wenn Sie eine Aufgabe starten. Wenn mehrere Dienste, die Service Discovery mit einer großen Anzahl von Aufgaben verwenden, gleichzeitig bereitgestellt werden, kann dies zu einer Überschreitung der AWS Cloud Map API-Drosselungsgrenzen führen. In diesem Fall werden Sie wahrscheinlich die folgende Meldung sehen: `Operations are being throttled. Will try again later` in Ihren Amazon ECS-Serviceereignismeldungen und langsamere Bereitstellungs- und Taskstartgeschwindigkeit. AWS Cloud Map dokumentiert keine Drosselungsgrenzen für diese APIs. Wenn Sie aufgrund dieser Einschränkungen eine Drosselung feststellen, können Sie sich an uns wenden, um Hilfe AWS Support zur Erhöhung Ihrer API-Drosselungsgrenzen zu erhalten. Weitere Empfehlungen zur Überwachung und Behebung solcher Drosselungsfehler finden Sie unter. Probleme mit der Drosselung von Amazon ECS lösen

Amazon-ECS-API-Referenz

Neben dem AWS Management Console und dem AWS Command Line Interface (AWS CLI) bietet Amazon ECS auch eine API. Sie können die API verwenden, um Aufgaben zur Verwaltung von Amazon-ECS-Ressourcen zu automatisieren.

- Eine Liste der API-Operationen nach Amazon-ECS-Ressourcen finden Sie unter Aktionen nach Amazon-ECS-Ressourcen.
- Eine alphabetische Liste der API-Operationen finden Sie unter Aktionen.
- Eine alphabetische Liste der Datentypen finden Sie unter Datentypen.
- Eine Liste der häufigen Abfrageparameter finden Sie unter Häufige Parameter.
- Beschreibungen der Fehlercodes finden Sie unter Häufige Fehler.

Weitere Informationen zu finden Sie in der AWS CLIAWS Command Line Interface Referenz für Amazon ECS.

Dokumentverlauf

Die folgende Tabelle beschreibt die wichtigen Updates und neuen Features für das Benutzerhandbuch für Amazon Elastic Container Service. Wir aktualisieren die Dokumentation regelmäßig, um das Feedback, das Sie uns senden, einzuarbeiten.

Änderung	Beschreibung	Datum
Unterstützung für GmSA für Linux-Container auf Fargate-Basis	Amazon ECS unterstützt die Active Directory-Authentifizierung für Linux-Container auf Fargate über ein spezielles Dienstkonto, das als Group Managed Service Account (gMSA) bezeichnet wird. Weitere Informationen finden Sie unter Verwenden gMSA für Linux Container auf Fargate.	5. März 2024
CloudWatch Metriken für Amazon EBS-Volumes hinzugefügt, die an Aufgaben angehängt sind	Amazon ECS veröffentlicht jetzt CloudWatch Metriken zur Amazon EBS-Speichernutzung für Aufgaben, denen ein Amazon EBS-Volume zugeordnet ist. Weitere Informationen finden Sie unter Amazon CloudWatch ECS-Metriken.	8. Februar 2024
Service Connect TLS	Sie können jetzt TLS mit Service Connect verwenden.	22. Januar 2024
Verwaltete TLS-Richtlinie für Service Connect	Neue InfrastructureRolePolicyForServiceConnectTransportLayerSecurityAmazonECS-Richtlinie hinzugefügt.	22. Januar 2024
Update der Service Connect-Timeout-Konfiguration	Die Service Connect-Timeout-Konfiguration kann jetzt aktualisiert werden und umfasst zwei optionale Parameter - `idleTimeout` und `perRequestTimeout`.	22. Januar 2024
Entleerung von verwalteten Amazon ECS-Instanzen	Sie können Amazon ECS Managed Instance Draining verwenden, um die ordnungsgemäße Kündigung von Amazon ECS-Instances zu ermöglichen.	19. Januar 2024

Änderung	Beschreibung	Datum
Ubuntu 2.2-Unterstützung für ECS Anywhere hinzugefügt	Support für das Betriebssystem Ubuntu 22 wurde zu ECS Anywhere hinzugefügt. Weitere Informationen finden Sie unter Unterstützte Betriebssysteme und Systemarchitekturen.	16. Januar 2024
`AmazonECSInfrastructureRolePolicyForVolumes` IAM-Richtlinie hinzufügen	Der `AmazonECSInfrastructureolePolicyForVolumes` wurde hinzugefügt. Die Richtlinie gewährt die Berechtigungen, die Amazon ECS benötigt, um AWS API-Aufrufe zur Verwaltung von Amazon EBS-Volumes im Zusammenhang mit Amazon ECS-Workloads durchzuführen.	11. Januar 2024
Amazon EBS-Datenvolumen für Amazon ECS-Aufgabe	Sie können während der Bereitstellung 1 Amazon EBS-Datenvolumen pro Aufgabe für die Zuordnung zu eigenständigen Amazon ECS-Aufgaben oder Aufgaben, die von einem ECS-Service verwaltet werden, konfigurieren. Durch die Konfiguration eines Volumes bei der Bereitstellung können Sie wiederverwendbare Aufgabendefinitionen erstellen, die nicht auf bestimmte Volumetypen oder Einstellungen beschränkt sind. Amazon EBS-Volumes bieten einen hochverfügbaren, kostengünstigen, langlebigen und leistungsstarken Blockspeicher für datenintensive containerisierte Workloads.	11. Januar 2024
Die klassische Amazon ECS-Konsole hat das Ende ihrer Lebensdauer erreicht	Die Amazon ECS-Konsole hat das Ende ihrer Lebensdauer erreicht.	4. Dezember 2023
Aktualisierte Richtlinie	Die von AmazonECS ServiceRolePolicy verwaltete IAM-Richtlinie wurde mit neuen `autoscaling` und `autoscaling-plans` zusätzlichen `events` Berechtigungen aktualisiert.	4. Dezember 2023

Änderung	Beschreibung	Datum
Unterstützung für Runtime Monitoring	Sie können Runtime Monitoring verwenden, um Ihre Amazon ECS-Workloads zu überwachen und böswillig es oder nicht autorisiertes Verhalten zu erkennen. Weitere Informationen finden Sie unter Runtime Monitoring.	26. November 2023
Aktualisierte Richtlinie	Die `AmazonECSServiceRolePolicy` verwaltete IAM-Richtlinie wurde aktualisiert, um den Zugriff auf die AWS Cloud Map `DiscoverInstancesRevision` API zu ermöglichen.	04. Oktober 2023
AWS Fargate Konfiguration der Außerbetriebnahme von Aufgaben	Sie können die Wartezeit konfigurieren, bevor Fargate-Aufgaben außer Betrieb genommen werden. Weitere Informationen finden Sie unter AWS Fargate - Aufgabenverwaltung.	5. September 2023
Zusätzliche Parameter für die Aufgabendefinition in AWS Fargate	AWS Fargate fügt Unterstützung für `pidMode` und `systemControls` in der Linux-Plattformversion hinzu`1.4.0`. Weitere Informationen finden Sie unter Aufgabendefinitionen.	9. August 2023
Neugestaltung der Aufgabendefinitionsseite der Amazon-ECS-Konsole	Die Aufgabendefinitionsseite in der Amazon-ECS-Konsole wurde neu gestaltet und enthält zusätzliche Optionen. Weitere Informationen finden Sie unter Erstellen einer Aufgabendefinition mithilfe der Konsole.	26. Juli 2023
Fargate unterstützt Lazy Loading mit Indizes für Seekable OCI	AWS Fargate führt Seekable OCI (SOCI) -Indizes ein. Mit SOCI verbringen Container nur wenige Sekunden mit dem Abruf von Images, bevor sie starten können. So bleibt Zeit für die Einrichtung der Umgebung und die Instanziierung der Anwendung, während das Image im Hintergrund heruntergeladen wird. Weitere Informationen finden Sie unter Lazy Loading Container Images using Seekable OCI (SOCI) im Amazon ECS-Benutzerhandbuch für Fargate. AWS	17. Juli 2023

Änderung	Beschreibung	Datum
Verbesserte Unterstützung für gMSA unter Linux und Windows	Die Aufgabendefinition enthält ein neues `credentialSpecs` -Feld für gMSA für Linux und Windows. Ein neues vollständiges Tutorial für domainlose gMSA unter Windows wurde hinzugefügt, siehe Tutorial: Verwenden von Windows Containern mit Domainlosen gMSA unter Verwendung von AWS CLI. Weitere Informationen finden Sie unter Verwenden von gMSAs für Linux Container und Verwenden von gMSAs für Windows Container.	14. Juli 2023
Verbesserte Dokumentation zu ECS-Agenten-Versionen	Die Dokumentation für die Amazon-ECS Agent-Versionen wurde aktualisiert. Wir empfehlen, dass Sie die Version `v20.10.13` oder eine neuere Version von Docker mit der neuesten Version des Amazon-ECS-Container-Agenten verwenden. Die veröffentlichten Versionen und Änderungen am Agenten sind verfügbar unter. GitHub Weitere Informationen finden Sie auf GitHub unter Versionen des Amazon ECS Linux Container Agenten.	20. Juni 2023
Aktualisierte regionale Verfügbarkeit für Fargate-ARM64-Unterstützung	Die regionale Verfügbarkeit für die ARM64-Unterstützung von Fargate wurde aktualisiert. Weitere Informationen finden Sie unter Überlegungen.	19. Juni 2023
Dokumentation für Verbesserung von Cluster-Auto-Scaling	Die Dokumentation zur Amazon-ECS-Skalierung von Amazon EC2 Auto Scaling wurde in Bezug auf Genauigkeit und Lesbarkeit erheblich verbessert. Weitere Informationen finden Sie unter Auto Scaling von Amazon-ECS-Clustern.	4. Mai 2023

Änderung	Beschreibung	Datum
Tagging-Autorisierung für die Erstellung von Ressourcen.	Benutzer müssen über Berechtigungen für Aktionen verfügen, die die Ressource erstellen, wie z. B. `ecsCreateCluster` . Wenn Sie eine Ressource erstellen und Tags für diese Ressource angeben, AWS führt eine zusätzliche Autorisierung durch, um zu überprüfen, ob Berechtigungen zum Erstellen von Tags vorhanden sind. Weitere Informationen finden Sie unter Tagging-Autorisierung und Berechtigung zum Taggen von Ressourcen bei der Erstellung gewähren.	18. April 2023
Support für gMSA für Linux-Container in EC2	Sie können gMSA verwenden, um sich bei Active Directory für Linux-Container auf EC2 zu authentif izieren. Weitere Informationen finden Sie unter Verwenden von gMSAs für Linux-Container.	14. April 2023
Support für flüchtigen Speicher für Windows-C ontainer in AWS Fargate	Sie können den flüchtigen Speicher für Windows-C ontainer in AWS Fargate verwenden. Weitere Informati onen finden Sie unter Fargate-Aufgabenspeicher.	14. April 2023
AWS Cost Managemen t Unterstützung für CUR-Daten auf Aufgabene bene	In den Kosten- und Nutzungsberichten können Sie Kosten und Ressourcenverbrauch auf Aufgabene bene aktivieren. Dadurch werden Split Cost Allocation Data für Aufgaben hinzugefügt, die auf AWS Fargate und EC2 ausgeführt werden. Weitere Informationen finden Sie unter Kosten- und Nutzungsberichte auf Aufgabenebene.	12. April 2023
Amazon-ECS-optimiertes Amazon-Linux-2023-AMI	Sie können Workloads auf dem Amazon-ECS-optimie rten Amazon-Linux-2023-AMI bereitstellen. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	10. April 2023

Änderung	Beschreibung	Datum
AWS Fargate Bundesstandard für Informationsverarbeitung (FIPS) 140	Sie können Workloads auf Amazon ECS auf eine AWS Fargate Weise bereitstellen, die dem Federal Information Processing Standard (FIPS) 140 entspricht. Weitere Informationen finden Sie unter AWS Fargate Bundesstandard für Informationsverarbeitung (FIPS-140).	10. April 2023
Löschen von Aufgabendefinitionen	Sie können eine Aufgabendefinition mithilfe der Amazon-ECS-Konsole, des SDK und der AWS CLI löschen. Weitere Informationen finden Sie unter Löschen einer Aufgabendefinitionsrevision mithilfe der Konsole und Aufgabendefinitionen.	24. Februar 2023
AWS Fargate Serviceempfehlungen in Compute Optimizer	AWS Compute Optimizer generiert Empfehlungen zur Aufgaben- und Containergröße auf der Grundlage der Nutzung laufender Aufgaben in Amazon ECS-Services auf AWS Fargate. Weitere Informationen finden Sie unter Anzeigen von Empfehlungen für Amazon-ECS-Services in Fargate.	27. Januar 2023
Amazon-ECS-Konsole	Die neue Amazon-ECS-Konsole ist jetzt die Standardkonsole. Weitere Informationen finden Sie unter Neue Amazon-ECS-Konsole.	19. Januar 2023
Aktualisierte `AmazonECS_FullAccess` -IAM-Richtlinie	Die `AmazonECS_FullAccess` -IAM-Richtlinie wurde aktualisiert und umfasst nun Berechtigungen zum Hinzufügen von Tags zu Load Balancern während der Erstellung. Weitere Informationen finden Sie unter AmazonECS_ FullAccess.	4. Januar 2023
Verwenden Sie CloudWatch Alarme, um Fehler bei der Bereitstellung von Amazon ECS-Services zu erkennen	Sie können Amazon ECS so konfigurieren, dass die Bereitstellung auf „Fehlgeschlagen“ gesetzt wird, wenn erkannt wird, dass ein bestimmter CloudWatch Alarm in den ALARM-Status übergegangen ist. Weitere Informationen finden Sie unter the section called “Erkennung von Fehlern ”.	19. Dezember 2022

Änderung	Beschreibung	Datum
Unterstützung für die Zuordnung von Container-Ports	Sie können einen Portnummernbereich für den Container festlegen, der an den dynamisch zugeordneten Host-Portbereich gebunden ist. Weitere Informationen finden Sie unter the section called “Port-Zuweisungen”.	15. Dezember 2022
Allgemeine Verfügbarkeit von Amazon ECS Service Connect	Diese Funktion fügt Service-Erkennung und Service-Mesh hinzu, die von Amazon-ECS-Service-Bereitstellungen gesteuert werden. Weitere Informationen finden Sie unter the section called “ Service Connect”.	27. November 2022
Die neue Amazon ECS-Konsolenerfahrung für Aufgabendefinitionen wurde aktualisiert	Das neue Amazon ECS-Konsolenerlebnis enthält jetzt einen JSON-Editor für Aufgabendefinitionen. Weitere Informationen finden Sie unter the section called “Erstellen einer Aufgabendefinition mit der Konsole”.	27. Oktober 2022
Die neue Amazon ECS-Konsolenerfahrung für Aufgabendefinitionen wurde aktualisiert	Das neue Amazon ECS-Konsolenerlebnis enthält jetzt einen JSON-Editor für Aufgabendefinitionen. Weitere Informationen finden Sie unter the section called “Erstellen einer Aufgabendefinition mit der Konsole”.	27. Oktober 2022
Die neue Amazon-ECS-Konsolenerfahrung wurde aktualisiert	Die neue Amazon ECS-Konsolenerfahrung wurde mit zusätzlichen Service- und Aufgabenparametern aktualisiert. Weitere Informationen finden Sie unter the section called “Erstellen eines Service” und the section called “Eine Anwendung als Aufgabe ausführen”.	7. Oktober 2022
Neue Informationen im Aufgabenmetadaten-Endpunkt Version 4	Der Aufgaben-Metadaten-Endpunkt Version 4 enthält jetzt die VPC-ID und den Servicenamen. Weitere Informationen finden Sie unter the section called “Aufgabenmetadaten-Endpunkt Version 4”.	7. Oktober 2022
Neue Größen der Aufgabendefinition	Amazon ECS in Fargate unterstützt jetzt die Aufgabengrößen 8 vCPU und 16 vCPU. Weitere Informationen finden Sie unter the section called “Aufgabengröße”.	16. September 2022

Änderung	Beschreibung	Datum
ECS-CLI-Seiten archiviert	Die ECS-CLI-Dokumentation wurde archiviert. Wir empfehlen, AWS Copilot für Ihre Anforderungen an Befehlszeilentools zu verwenden. Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.	15. September 2022
Neue Fargate-Kontingente	Fargate wird von aufgabenanzahlbasierten Kontingenten auf vCPU-basierte Kontingente umgestellt. Weitere Informationen finden Sie unter the section called “AWS Fargate Servicekontingenten”.	8. September 2022
Support für Warm-Pools von Amazon EC2 Auto Scaling.	Sie können jetzt mit Warm-Pools von Amazon EC2 Auto Scaling Ihre Anwendungen schneller skalieren und beim Preis sparen. Weitere Informationen finden Sie unter Konfiguration vorinitialisierter Instances für Ihre Amazon ECS Auto Scaling Scaling-Gruppe.	23. März 2022
Support für Windows-Instances in ECS Anywhere.	ECS Anywhere unterstützt jetzt Windows-Instances. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den externen Starttyp.	3. März 2022
ECS-Exec-Support für externe Instances hinzugefügt	ECS Exec wird jetzt für externe Instances unterstützt. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit ECS Exec.	24. Januar 2022
Das neue Amazon-ECS-Konsolenerlebnis wurde aktualisiert	Das neue Erlebnis der Amazon-ECS-Konsole unterstützt das Erstellen und Löschen eines Clusters, das Aktualisieren einer Aufgabendefinition und das Abmelden einer Aufgabendefinition. Weitere Informationen dazu finden Sie unter Erstellen eines Amazon ECS-Clusters für den Starttyp Fargate, Löschen eines Amazon ECS-Clusters, Aktualisieren einer Amazon ECS-Aufgabendefinition mithilfe der Konsole und Abmeldung einer Revision der Amazon ECS-Aufgabendefinition mithilfe der Konsole.	8. Dezember 2021

Änderung	Beschreibung	Datum
Das neue Amazon-ECS-Konsolenerlebnis wurde aktualisiert	Das neue Amazon-ECS-Konsolenerlebnis unterstützt das Erstellen einer Aufgabendefinition. Weitere Informationen finden Sie unter Erstellen einer Amazon ECS-Aufgabendefinition mithilfe der Konsole.	23. November 2021
Amazon ECS unterstützt die 64-Bit-ARM-Architektur für Linux.	Amazon ECS unterstützt die 64-Bit-ARM-CPU-Architektur für das Linux-Betriebssystem. Weitere Informationen finden Sie unter the section called “Aufgabendefinitionen für 64-Bit-ARM-Workloads”.	23. November 2021
Amazon ECS-Unterstützung für die Fluentd-Option log-driver-buffer-limit	Amazon ECS unterstützt die fluentd-Option `log-driver-buffer-limit` . Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.	22. November 2021
Amazon-ECS-optimiertes Linux-AMI-Entwicklungsskript	Amazon ECS hat die Entwicklungsskripts, die zum Erstellen der Linux-Varianten des Amazon-ECS-optimierten AMI verwendet werden, als Open Source bereitgestellt. Weitere Informationen finden Sie unter Amazon-ECS-optimiertes Linux-AMI-Entwicklungsskript.	19. November 2021
Zustand der Container-Instance	Amazon ECS fügt Support für die Zustandsüberwachung von Container-Instances hinzu. Weitere Informationen finden Sie unter Überwachen Sie den Zustand der Amazon ECS-Container-Instance.	10. November 2021
Amazon-ECS-Exec-Support für Windows	Amazon ECS Exec unterstützt Windows. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit ECS Exec.	1. November 2021
Support für Windows-Container auf Fargate.	Amazon ECS unterstützt Windows-Container auf Fargate. Weitere Informationen finden Sie unter Fargate Windows-Plattformversionen für Amazon ECS.	28. Oktober 2021

Änderung	Beschreibung	Datum
GPU-Support für externe Instances auf Amazon ECS Anywhere	Amazon ECS unterstützt die Angabe von GPU-Anforderungen in der Aufgabendefinition für Aufgaben, die auf externen Instances ausgeführt werden. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads und Registrierung einer externen Instance in einem Amazon ECS-Cluster.	8. Oktober 2021
Support von `awsvpc`-Netzwerkmodus unter Windows	Amazon ECS unterstützt `awsvpc`-Netzwerkmodus unter Windows. Weitere Informationen finden Sie unter Zuweisen einer Netzwerkschnittstelle für eine Amazon ECS-Aufgabe.	15. Juli 2021
Allgemeine Verfügbarkeit von Bottlerocket	Amazon ECS unterstützt eine Amazon-ECS-optimierte AMI-Variante des Betriebssystems Bottlerocket und wird als AMI zur Verfügung gestellt. Weitere Informationen finden Sie unter Amazon-ECS-optimierte Bottlerocket-AMIs.	30. Juni 2021
Aktualisieren geplanter Aufgaben von Amazon ECS	Amazon EventBridge hat Unterstützung für zusätzliche Parameter bei der Erstellung von Regeln hinzugefügt, die geplante Amazon ECS-Aufgaben auslösen.	25. Juni 2021
AWS verwaltete Richtlinien für Amazon ECS	Amazon ECS hat eine Dokumentation der AWS verwalteten Richtlinien für serviceverknüpfte Rollen hinzugefügt. Weitere Informationen finden Sie unter AWS verwaltete Richtlinien für Amazon Elastic Container Service.	8. Juni 2021
Erste Schritte mit dem AWS CDK	Es wurde eine Anleitung für die ersten Schritte zur Verwendung von AWS CDK mit Amazon ECS hinzugefügt. Weitere Informationen finden Sie unter Erstellen von Amazon ECS-Ressourcen mit dem AWS CDK.	27. Mai 2021

Änderung	Beschreibung	Datum
Amazon ECS Anywhere	Amazon ECS hat Unterstützung für die Registrierung eines lokalen Servers oder einer virtuellen Maschine (VM) in Ihrem Cluster hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den externen Starttyp.	25. Mai 2021
Amazon-ECS-optimiertes Windows Server 20H2 Core AMI	Amazon ECS hat Unterstützung für eine neue Windows Amazon-ECS-optimierte AMI-Variante für Windows Server 20H2 Core hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	19. April 2021
Amazon ECS Exec	Amazon ECS hat ein neues Debugging-Tool namens ECS Exec veröffentlicht. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit ECS Exec.	15. März 2021
Support der VPC-Endpunkt-Richtlinie	Amazon ECS unterstützt jetzt VPC-Endpunktrichtlinien. Weitere Informationen finden Sie unter Erstellen einer VPC-Endpunktrichtlinie für Amazon ECS.	11. Januar 2021
Neue -Konsolenumgebung	Amazon ECS hat eine neue Konsolenerfahrung veröffentlicht, die das Erstellen oder Aktualisieren eines Dienstes oder das Ausführen einer eigenständigen Aufgabe unterstützt. Weitere Informationen finden Sie unter Einen Amazon ECS-Service mithilfe der Konsole erstellen und Eine Anwendung als Amazon ECS-Aufgabe ausführen.	28. Dezember 2020
Aktualisierung des Kapazitätsanbieters	Amazon ECS hat Unterstützung für das Aktualisieren eines vorhandenen Auto-Scaling-Gruppenkapazitätsanbieters hinzugefügt.	23. November 2020

Änderung	Beschreibung	Datum
ECS unterstützt jetzt Amazon FSx for Windows File Server für Windows Aufgaben	Amazon ECS hat Unterstützung für die Angabe von Amazon FSx for Windows File Server Volumes in Windows-Task-Definitionen hinzugefügt. Weitere Informationen finden Sie unter Verwenden Sie FSx for Windows File Server Windows-Dateiserver-Volumes mit Amazon ECS.	11. November 2020
Unterstützung für VPC Dual-Stack-Modus hinzugefügt	Amazon ECS hat Unterstützung für die Verwendung einer VPC im Dual-Stack-Modus mit Aufgaben hinzugefügt, die den `awsvpc`-Netzwerkmodus, der IPv6-Adressen unterstützt, nutzen. Weitere Informationen finden Sie unter Verwenden einer VPC im Dual-Stack-Modus.	5. November 2020
Aktualisieren von Aufgabenmetadaten-Endpunkt v4	Amazon ECS hat zusätzliche Metadaten zur Ausgabe des Endpunkts v4 für die Task-Metadaten hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4.	5. November 2020
Support für Local Zones und Wavelength Zones	Amazon ECS hat Unterstützung für Workloads in Local Zones und Wavelength Zones hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-Anwendungen in gemeinsam genutzten Subnetzen, Local Zones und Wellenlängenzonen.	4. September 2020
Amazon-ECS-Variante von Bottlerocket AMI	Bottlerocket ist ein Linux-basiertes Open-Source-Betriebssystem, das speziell für den Betrieb von Containern entwickelt wurde. AWS Eine Amazon-ECS-optimierte AMI-Variante des Betriebssystems Bottlerocket wird als AMI bereitgestellt, die Sie beim Starten von Amazon-ECS-Container-Instances verwenden können. Weitere Informationen finden Sie unter Amazon-ECS-optimierte Bottlerocket-AMIs.	31. August 2020

Änderung	Beschreibung	Datum
Aktualisierung der Aufgabenmetadaten-Endpunkt Version 4 für Netzwerkratenstatistiken	Die Aufgabenmetadaten Endpunkt Version 4 wurde aktualisiert, um Netzwerkratenstatistiken für Amazon-ECS-Aufgaben bereitzustellen, die die `awsvpc`- oder `bridge`-Netzwerkmodi, die auf Amazon EC2 Instances, auf denen mindestens Version `1.43.0` des Container-Agenten gehostet werden, nutzen. Weitere Informationen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4.	10. August 2020
Fargate-Nutzungskennzahlen	AWS Fargate bietet CloudWatch Nutzungsmetriken, die Einblick in die Nutzung der Fargate On-Demand - und Fargate Spot-Ressourcen durch Ihre Konten geben. Weitere Informationen finden Sie unter Nutzungsmetriken.	3. August 2020
AWS Copilot Version 0.1.0	Die neue AWS Copilot-CLI wurde eingeführt und bietet Befehle auf hoher Ebene, um die Modellierung, Erstellung, Veröffentlichung und Verwaltung von containerisierten Anwendungen auf Amazon ECS von einer lokalen Entwicklungsumgebung aus zu vereinfachen. Weitere Informationen finden Sie unter Amazon ECS-Ressourcen mithilfe der AWS Copilot-Befehlszeilenschnittstelle erstellen.	9. Juli 2020
AWS Fargate Zeitplan für die Vernachlässigung von Plattformversionen	Der Zeitplan für die Veraltung der Fargate Plattformversion wurde hinzugefügt. Weitere Informationen finden Sie unter AWS Veraltete Version der Fargate-Linux-Plattform.	8. Juli 2020
AWS Erweiterung der Region Fargate	Amazon ECS on AWS Fargate wurde auf die Region Europa (Mailand) ausgeweitet.	25. Juni 2020

Änderung	Beschreibung	Datum
Amazon-ECS-optimiertes AMI für Amazon Linux 2 (Neuron) veröffentlicht	Amazon ECS hat ein Amazon-ECS-optimiertes AMI für Amazon Linux 2 (Neuron) für inferentielle Workloads veröffentlicht. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	24. Juni 2020
Unterstützung für das Löschen von Kapazität sanbietern wurde hinzugefügt	Amazon ECS hat Unterstützung für das Löschen von Auto Scaling-Gruppenkapazitätsanbietern hinzugefügt.	11. Juni 2020
AWS Aktualisierung der Fargate-Plattform auf Version 1.4.0	Ab dem 28. Mai 2020 wird bei jeder neuen Fargate-Aufgabe, die mit der Plattformversion 1.4.0 gestartet wird, der kurzlebige 20-GB-Speicher mit einem AES-256-Verschlüsselungsalgorithmus unter Verwendung eines von AWS Fargate verwalteten Verschlüsselungsschlüssels verschlüsselt. Weitere Informationen finden Sie unter Flüchtiger Speicher für Fargate-Aufgaben für Amazon ECS.	28. Mai 2020
Support für Umgebungs variablendatei	Support für das Angeben von Umgebungsvariablen dateien in einer Aufgabendefinition hinzugefügt, mit der Sie Ihren Containern Umgebungsvariablen gesammelt hinzufügen können. Weitere Informati onen finden Sie unter Übergeben Sie eine einzelne Umgebungsvariable an einen Amazon ECS-Container.	18. Mai 2020
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Region Afrika (Kapstadt) ausgeweitet.	11. Mai 2020

Änderung	Beschreibung	Datum
Aktualisierung des Servicekontingents	Das folgende Servicekontingent wurde aktualisiert: • Die Zahl der Cluster pro Konto wurde von `2,000` auf `10,000` erhöht. Weitere Informationen finden Sie unter Amazon-ECS-Service-Kontingente.	17. April 2020

Änderung	Beschreibung	Datum
AWS Fargate-Plattformversion 1.4.0	AWS Die Version 1.4.0 der Fargate-Plattform wird veröffentlicht, die die folgenden Funktionen enthält: • Unterstützung für die Verwendung von Amazon EFS Dateisystem-Volumes für die persistente Aufgabenspeicherung hinzugefügt. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS. • Der kurzlebige Aufgabenspeicher wurde auf 20 GB erhöht. Weitere Informationen finden Sie unter Flüchtiger Speicher für Fargate-Aufgaben für Amazon ECS. • Das Verhalten des zu Aufgaben hinführenden und von Aufgaben wegführenden Netzwerkdatenverkehrs wurde aktualisiert. Ab Plattformversion 1.4 erhalten alle Fargate-Aufgaben eine einzige Elastic-Network-Schnittstelle (als Aufgaben-ENI bezeichnet) und der gesamte Netzwerkverkehr fließt innerhalb Ihrer VPC durch diese ENI und wird durch Ihre VPC-Flow-Protokolle für Sie sichtbar. Weitere Informationen finden Sie unter Fargate-Aufgabenvernetzung im Benutzerhandbuch zum Amazon Elastic Container Service für AWS Fargate. • Aufgaben-ENIs fügen Unterstützung für Jumbo-Frames hinzu. Netzwerkschnittstellen sind mit einer Maximum Transmission Unit (MTU) konfiguriert, die der Größe der größten Nutzlast entspricht, die in einen einzelnen Frame passt. Je größer die MTU ist, desto mehr Anwendungsnutzlast passt in ein einzelnes Frame, was den Overhead pro Frame reduziert und die Effizienz erhöht. Die Unterstüt	8. April 2020

Änderung	Beschreibung	Datum
	zung von Jumbo-Frames reduziert den Overhead, wenn der Netzwerkpfad zwischen Ihrer Aufgabe und dem Ziel Jumbo-Frames unterstützt, wie z. B. den gesamten Datenverkehr, der in Ihrer VPC verbleibt. • CloudWatch Container Insights wird Netzwerkl eistungskennzahlen für Fargate-Aufgaben enthalten. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights. • Unterstützung für den Aufgabenmetadaten-Endpunkt v4 hinzugefügt, der zusätzliche Informati onen für Ihre Fargate-Aufgaben enthält, einschlie ßlich Netzwerkstatistiken und in welcher Availability Zone die Aufgabe ausgeführt wird. Weitere Informati onen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4. • Unterstützung für den Linux-ParameterSYS_PTRAC E in Containerdefinitionen hinzugefügt. Weitere Informationen finden Sie unter Linux-Parameter. • Der Fargate Container-Agent ersetzt die Verwendun g des Amazon-ECS-Container-Agents für alle Fargate-Aufgaben. Diese Änderung sollte keine Auswirkungen auf die Ausführung Ihrer Aufgaben haben. • Die Containerlaufzeit verwendet nun Container d anstelle von Docker. Diese Änderung sollte keine Auswirkungen auf die Ausführung Ihrer Aufgaben haben. Sie werden feststellen, dass einige Fehlermeldungen, die in der Container laufzeit entstehen, jetzt nicht mehr Docker, sondern allgemeinere Fehlern erwähnen.	

Änderung	Beschreibung	Datum
	Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.	
Amazon EFS-Datei systemunterstützung für Aufgaben-Volumes	Amazon EFS-Dateisysteme können als Daten-Volumes sowohl für Ihre Amazon-ECS- als auch Ihre Fargate-Aufgaben verwendet werden. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.	8. April 2020
Amazon-ECS-Aufgabe nmetadaten-Endpunkt Version 4	Ab Amazon-ECS-Container-Agent-Version 1.39.0 und Fargate-Plattformversion 1.4.0 wird in jeden Container einer Aufgabe eine Umgebungsvariable mit dem Namen `ECS_CONTAINER_METADATA_URI_V4` injiziert. Wenn Sie den Aufgabenmetadaten-Endpunkt Version 4 abfragen, stehen für die Aufgaben verschied ene Aufgabenmetadaten und Docker-Statistiken zur Verfügung. Weitere Informationen finden Sie unter Amazon ECS-Endpunkt für Aufgabenmetadaten, Version 4.	8. April 2020
Hinzufügung von Unterstützung für bestimmte Versionen von Secrets Manager-Secrets, die als Umgebungsvariablen eingefügt werden sollen	Hinzufügung von Unterstützung für die Angabe vertraulicher Daten mit bestimmten Versionen von Secrets Manager-Secrets. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.	24. Februar 2020
Zusätzliche CodeDeplo y Einrichtungskonfig urationsoptionen für Blau/ Grün-Bereitstellungen hinzugefügt	Der CodeDeploy Service fügte neue kanarische und lineare Bereitstellungskonfigurationen für den Bereitstellungstyp Amazon ECS hinzu. Es können auch benutzerdefinierte Bereitstellungskonfiguratio nen definiert werden. Weitere Informationen finden Sie unter Überprüfen Sie den Status eines Amazon ECS-Service vor der Bereitstellung .	6. Februar 2020

Änderung	Beschreibung	Datum
Der Parameter für die efsVolumeConfiguration Aufgabendefinition wurde hinzugefügt	Der Aufgabendefinitionsparameter `efsVolume Configuration` befindet sich in der öffentlichen Vorschau, wodurch die Verwendung von Amazon EFS-Dateisystemen für Ihre Amazon-ECS-Aufgaben vereinfacht wird. Weitere Informationen finden Sie unter Verwenden Sie Amazon EFS-Volumes mit Amazon ECS.	17. Januar 2020
Aktualisierung des Verhaltens der Amazon-ECS-Container-Agent-Protokollierung	Die Protokollierungsorte und das Rotationsverhalten des Amazon-ECS-Container-Agenten wurden aktualisiert. Weitere Informationen finden Sie unter Konfigurationsparameter für das Amazon ECS-Container-Agent-Protokoll.	13. Januar 2020
Fargate Spot	Amazon ECS hat eine Unterstützung für die Ausführung von Aufgaben mit Fargate Spot hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-Cluster für den Starttyp Fargate .	3. Dezember 2019
Cluster Auto Scaling	Die Amazon ECS Cluster Auto Scaling bietet Ihnen mehr Kontrolle über die Skalierung von Aufgaben innerhalb eines Clusters. Weitere Informationen finden Sie unter Automatische Verwaltung der Amazon ECS-Kapazität mit Cluster-Auto-Scaling.	3. Dezember 2019
Cluster-Kapazitätsanbieter	Die Amazon-ECS-Cluster-Kapazitätsanbieter bestimmen die Infrastruktur, die für Ihre Aufgaben verwendet werden soll. Weitere Informationen finden Sie unter Amazon-ECS-Cluster.	3. Dezember 2019
Erstellen eines Clusters auf einem AWS Outposts	Amazon ECS unterstützt jetzt das Erstellen von Clustern auf einem AWS Outposts. Weitere Informationen finden Sie unter the section called “Amazon Elastic Container Service auf AWS Outposts”.	3. Dezember 2019

Änderung	Beschreibung	Datum
Service-Aktionsereignisse	Amazon ECS sendet jetzt Ereignisse an Amazon EventBridge , wenn bestimmte Serviceaktionen auftreten. Weitere Informationen finden Sie unter Aktionsereignisse des Amazon ECS-Service.	25. November 2019
Amazon ECS GPU-optimiertes AMI unterstützt G4-Instances	Amazon ECS hat Unterstützung für die g4-Instance-Typ-Familie bei Verwendung des Amazon ECS GPU-optimierten AMI hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.	8. Oktober 2019
FireLens für Amazon ECS	FireLens für Amazon ECS ist allgemein verfügbar . FireLens für Amazon ECS können Sie Aufgabendefinitionsparameter verwenden, um Protokolle zur Protokollspeicherung und Analyse an ein AWS Service- oder Partnerziel weiterzuleiten. Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.	30. September 2019
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Regionen Europa (Paris), Europa (Stockholm) und Naher Osten (Bahrain) ausgeweitet.	30. September 2019
Deep Learning Containers mit Elastic Inference auf Amazon ECS	Amazon ECS unterstützt das Anfügen von Amazon Elastic Inference-Accelerators an Ihre Container , um die Ausführung von Deep Learning-Inferenz-Workloads effizienter zu gestalten. Weitere Informationen finden Sie unter Deep Learning Containers mit Elastic Inference auf Amazon ECS.	03. September 2019

Änderung	Beschreibung	Datum
FireLens für Amazon ECS	FireLens für Amazon ECS befindet sich in der öffentlichen Vorschauversion. FireLens für Amazon ECS können Sie Aufgabendefinitionsparameter verwenden, um Protokolle zur Protokollspeicherung und Analyse an ein AWS Service- oder Partnerziel weiterzuleiten. Weitere Informationen finden Sie unter Amazon ECS-Protokolle an einen AWS Service senden oder AWS Partner.	30. August 2019
CloudWatch Einblicke in Container	CloudWatch Container Insights ist jetzt allgemein verfügbar. Es ermöglicht Ihnen, Metriken und Protokolle aus Ihren containerisierten Anwendungen und Microservices zu sammeln, zu aggregieren und zusammenzufassen. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.	30. August 2019
Auslagerungskonfiguration auf Container ebene	Amazon ECS bietet nun Unterstützung für die Kontrolle der Nutzung von Auslagerungsspeicherplatz auf Ihren Linux-Container-Instances auf Container-Ebene. Bei der Verwendung einer Auslagerungskonfiguration pro Container kann die Auslagerung für jeden Container innerhalb einer Aufgabendefinition aktiviert oder deaktiviert sein. Für diejenigen, für die sie aktiviert ist, kann die maximale Menge des verwendeten Auslagerungsbereichs begrenzt sein. Weitere Informationen finden Sie unter Verwaltung des Container-Swap-Speicherplatzes auf Amazon ECS.	16. August 2019
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Region Asien-Pazifik (Hongkong) ausgeweitet.	6. August 2019

Änderung	Beschreibung	Datum
Elastic-Network-Schnittstellen-Trunking	Zusätzliche unterstützte Amazon-EC2-Instance-Typen für ENI-Trunking-Funktion hinzugefügt. Weitere Informationen finden Sie unter Unterstützte Instances für mehr Amazon ECS-Container-Netzwerkschnittstellen.	1. August 2019
Registrieren von mehreren Zielgruppen mit einem Service	Support für das Festlegen mehrerer Zielgruppen in einer Servicedefinition hinzugefügt. Weitere Informationen finden Sie unter Registrierung mehrerer Zielgruppen bei einem Amazon ECS-Service.	30. Juli 2019
Angeben vertraulicher Daten mithilfe von Secrets Manager-Secrets	Tutorial für das Angeben vertraulicher Daten mithilfe von Secrets Manager-Secrets wurde hinzugefügt. Weitere Informationen finden Sie unter Angabe sensibler Daten mithilfe von Secrets Manager Manager-Geheimnissen in Amazon ECS.	20. Juli 2019
CloudWatch Einblicke in Container	Amazon ECS hat Unterstützung für CloudWatch Container Insights hinzugefügt. Weitere Informationen finden Sie unter Überwachen Sie Amazon ECS-Container mit Container Insights.	9. Juli 2019
Berechtigungen auf Ressourcenebene für Amazon-ECS-Services und -Aufgabensätze	Amazon ECS verfügt über erweiterte Unterstützung für Berechtigungen auf Ressourcenebene für Amazon-ECS-Services und -Aufgaben. Weitere Informationen finden Sie unter So funktioniert Amazon Elastic Container Service mit IAM.	27. Juni 2019
Neues Amazon ECS-optimiertes AMI für -2019-005 gepatcht AWS	Amazon ECS hat das Amazon-ECS-optimierte AMI aktualisiert, um die in AWS-2019-005 beschriebenen Schwachstellen zu beheben.	17. Juni 2019

Änderung	Beschreibung	Datum
Elastic-Network-Schnittstellen-Trunking	Amazon ECS bietet jetzt Unterstützung für das Starten von Container-Instances mit unterstützten Amazon-EC2-Instance-Typen, die eine erhöhte Elastic-Network-Schnittstellen-Dichte aufweisen. Mit diesen Instance-Typen und einem Opt-in bei den `awsvpcTrunking` -Kontoeinstellungen erhalten Sie eine erhöhte ENI-Dichte bei neu gestarteten Container-Instances, sodass Sie mehr Aufgaben auf den einzelnen Container-Instances platzieren können. Weitere Informationen finden Sie unter Zunehmende Netzwerkschnittstellen für Amazon ECS Linux-Container-Instances.	6. Juni 2019
AWS Aktualisierung der Fargate-Plattform auf Version 1.3.0	Ab dem 1. Mai 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, den `splunk` -Protokolltreiber zusätzlich zum `awslogs`-Protokolltreiber. Weitere Informationen finden Sie unter Speicher und Protokollierung.	1. Mai 2019
AWS Aktualisierung der Fargate-Plattform auf Version 1.3.0	Ab dem 1. Mai 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, das Verweisen auf sensible Daten in der Protokollkonfiguration eines Containers mithilfe der `secretOptions` -Containerdefinitionsparameter. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.	1. Mai 2019

Änderung	Beschreibung	Datum
AWS Aktualisierung der Fargate-Plattform auf Version 1.3.0	Ab dem 2. April 2019 unterstützt jede neue Fargate-Aufgabe, die gestartet wird, das Injizieren sensibler Daten in Ihre Container, indem Ihre sensiblen Daten entweder in AWS Secrets Manager Secrets- oder AWS Systems Manager Parameter Store-Parametern gespeichert und dann in Ihrer Container-Definition referenziert werden. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.	2. Apr 2019
AWS Aktualisierung der Fargate-Plattform auf Version 1.3.0	Ab dem 27. März 2019 bietet jede neue gestartete Fargate-Aufgabe zusätzliche Aufgabendefinitionsparameter, mit denen Sie eine Proxy-Konfiguration, Abhängigkeiten für das Startup und Herunterfahren von Containern sowie einen Timeout-Wert für das Starten und Stoppen pro Container definieren können. Weitere Informationen finden Sie unter Proxykonfiguration, Container-Abhängigkeit und Container-Timeouts.	27. März 2019
Amazon ECS führt den externen Bereitstellungstyp ein	Der externe Bereitstellungstyp ermöglicht Ihnen, für die vollständige Kontrolle über den Bereitstellungsprozess für einen Amazon-ECS-Service einen beliebigen Drittanbieter-Bereitstellungs-Controller zu verwenden. Weitere Informationen finden Sie unter Stellen Sie Amazon ECS-Services mithilfe eines Controllers eines Drittanbieters bereit.	27. März 2019

Änderung	Beschreibung	Datum
AWS Deep Learning Containers auf Amazon ECS	AWS Deep Learning Containers sind eine Reihe von Docker-Images zum Trainieren und Bereitstellen von Modellen TensorFlow auf Amazon Elastic Container Service (Amazon ECS). Deep Learning Containers bieten optimierte Umgebungen mit TensorFlow Nvidia CUDA-Bibliotheken (für GPU-Instances) und Intel MKL (für CPU-Instances) und sind in Amazon ECR verfügbar. Weitere Informationen finden Sie unter Verwenden von AWS Deep Learning Containers auf Amazon ECS.	27. März 2019
Amazon ECS führt eine verbesserte Verwaltung von Container-Abhängigkeiten ein	Amazon ECS führt zusätzliche Aufgabendefinition sparameter ein, mit denen Sie Abhängigkeiten für das Starten und Herunterfahren von Containern sowie einen containerbezogenen Zeitüberschreitungswert für das Starten und Anhalten definieren können. Weitere Informationen finden Sie unter Container-Abhängigkeit.	7. März 2019
Amazon ECS führt die `PutAccountSettingDefault` -API ein	Amazon ECS führt die `PutAccountSettingDefault` -API ein, mit der ein Benutzer den Standard-Opt-in-Status für das ARN-/ID-Format für alle Benutzer und Rollen im Konto festlegen kann. Bisher erforderte das Festlegen des Standard-Opt-in-Status des Kontos die Verwendung des Kontobesitzers. Weitere Informationen finden Sie unter Amazon Ressourcennamen (ARNs) und IDs.	8. Februar 2019

Änderung	Beschreibung	Datum
Amazon ECS unterstützt GPU-Workloads	Amazon ECS führt die Unterstützung von GPUs-Workloads ein, indem Ihnen ermöglicht wird, Cluster mit GPU-fähigen Container-Instances zu erstellen. In einer Aufgabendefinition können Sie die Anzahl der erforderlichen GPUs angeben. Der ECS-Agent heftet dann die physischen GPUs an den Container an. Weitere Informationen finden Sie unter Amazon ECS-Aufgabendefinitionen für GPU-Workloads.	4. Februar 2019
Amazon ECS erweitert die Unterstützung von Secrets	Amazon ECS hat die Unterstützung für die direkte Verwendung von AWS Secrets Manager Geheimnissen in Ihren Aufgabendefinitionen erweitert, um sensible Daten in Ihre Container einzuschleusen. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.	21. Januar 2019
Schnittstellen-VPC-Endpunkte (AWS PrivateLink)	Zusätzliche Unterstützung für die Konfiguration von Schnittstellen-VPC-Endpunkten, die von AWS PrivateLink unterstützt werden. Dies ermöglicht Ihnen, eine private Verbindung zwischen Ihrer VPC und Amazon ECS zu erstellen, ohne dass ein Zugriff über das Internet, über eine NAT-Instance, eine VPN-Verbindung oder AWS Direct Connect erforderlich ist. Weitere Informationen finden Sie unter VPC-Schnittstellenendpunkte (AWS PrivateLink).	26. Dezember 2018

Änderung	Beschreibung	Datum
AWS Fargate-Plattformversion 1.3.0	Neue Version der AWS Fargate-Plattform veröffentlicht, die Folgendes enthält: • Unterstützung für die Verwendung von AWS Systems Manager Parameter Store-Parametern zum Einfügen sensibler Daten in Ihre Container wurde hinzugefügt. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container. • Aufgabenrecycling für Fargate-Aufgaben hinzugefügt. Dies ist der Prozess der Aktualisierung von Aufgaben, die Teil eines Amazon-ECS-Service sind. Weitere Informationen finden Sie unter Aufgaben-Wartung im Benutzerhandbuch zum Amazon Elastic Container Service für AWS Fargate. Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.	17. Dezember 2018

Änderung	Beschreibung	Datum
Service Limits aktualisiert	Die folgenden Service Limits wurden aktualisiert: • Die Anzahl Cluster pro Region und Konto wurde von `1000` auf `2000` erhöht. • Die Anzahl Container-Instances pro Cluster wurde von `1000` auf `2000` erhöht. • Die Anzahl Services pro Cluster wurde von `500` auf `1000` erhöht. Weitere Informationen finden Sie unter Amazon-ECS-Service-Kontingente.	14. Dezember 2018
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Regionen Asien-Pazifik (Mumbai) und Kanada (Zentral) ausgeweitet. Weitere Informationen finden Sie unter Unterstützte Regionen für Amazon ECS auf AWS Fargate.	7. Dezember 2018
Amazon-ECS-Blau/Grün-Bereitstellungen	Amazon ECS hat Unterstützung für blaue/grüne Bereitstellungen hinzugefügt mit. CodeDeploy Mit diesem Bereitstellungstyp können Sie eine neue Bereitstellung eines Services überprüfen, bevor Sie den Produktionsverkehr an ihn senden. Weitere Informationen finden Sie unter Überprüfen Sie den Status eines Amazon ECS-Service vor der Bereitstellung .	27. November 2018

Änderung	Beschreibung	Datum
Amazon-ECS-optimiertes Amazon Linux 2 (arm64) AMI veröffentlicht	Amazon ECS veröffentlichte eine Amazon-ECS-optimierte Amazon Linux 2-AMIs für die arm64-Architektur. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	26. November 2018
Unterstützung für zusätzliche Docker-Flags in den Aufgabendefinitionen hinzugefügt	Für Amazon ECS wurde die Unterstützung für die folgenden Docker-Flags in Aufgabendefinitionen eingeführt: • IPC-Modus • PID-Modus	16. November 2018
Support für Amazon-ECS-Secrets	Amazon ECS hat Unterstützung für die Verwendung von AWS Systems Manager Parameter Store-Parametern hinzugefügt, um sensible Daten in Ihre Container einzufügen. Weitere Informationen finden Sie unter Übergeben Sie sensible Daten an einen Amazon ECS-Container.	15. November 2018
Ressourcen-Markierung	Amazon ECS hat Unterstützung für das Hinzufügen von Metadaten-Tags zu Ihren Services, Aufgabendefinitionen, Aufgaben, Clustern und Container-Instances hinzugefügt. Weitere Informationen finden Sie unter Verschlagwortung von Amazon ECS-Ressourcen.	15. November 2018

Änderung	Beschreibung	Datum
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Regionen USA West (Nordkalifornien) und Asien-Pazifik (Seoul) ausgeweitet. Weitere Informationen finden Sie unter AWS Fargate für Amazon ECS.	7. November 2018
Service Limits aktualisiert	Die folgenden Service Limits wurden aktualisiert: • Die Anzahl der Aufgaben, die den Starttyp Fargate pro Region und pro Konto verwenden, wurde von 20 auf 50 erhöht. • Die Anzahl der öffentlichen IP-Adressen für Aufgaben, die den Fargate-Launch-Typ verwenden, wurde von 20 auf 50 erhöht. Weitere Informationen finden Sie unter Amazon-ECS-Service-Kontingente.	31. Oktober 2018
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Region Europa (London) ausgeweitet. Weitere Informationen finden Sie unter AWS Fargate für Amazon ECS.	26. Oktober 2018

Änderung	Beschreibung	Datum
Amazon-ECS-optimiertes Amazon Linux 2 AMI veröffentlicht	Amazon ECS bietet Linux-AMIs in zwei Varianten an, die für den Service optimiert sind. Die neueste und empfohlene Version basiert auf x;. Amazon ECS verkauft auch AMIs, die auf dem basieren, aber wir empfehlen Ihnen, Ihre Workloads auf die Amazon Linux 2-Variante zu migrieren, da der Support für das Amazon Linux AMI spätestens am 30. Juni 2020 endet. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	18. Oktober 2018
Amazon-ECS-Aufgabenmetadaten-Endpunkt Version 3	Ab Version 1.21.0 des Amazon-ECS-Containeragenten führt der Agent eine Umgebungsvariable namens `ECS_CONTAINER_METADATA_URI` in jeden Container einer Aufgabe ein. Wenn Sie den Endpunkt der Aufgaben-Metadaten Version 3 abfragen, stehen verschiedene Aufgaben-Metadaten und Docker-Statistiken für Aufgaben zur Verfügung, die den Netzwerkmodus `awsvpc` an einem HTTP-Endpunkt verwenden, der vom Amazon-ECS-Containeragenten bereitgestellt wird. Weitere Informationen finden Sie unter Überwachen Sie Workloads mithilfe von Amazon ECS-Metadaten.	18. Oktober 2018
Erweiterung der Regionen für Amazon-ECS-Service Discovery	Amazon-ECS-Service Discovery bietet erweiterte Unterstützung in den Regionen Kanada (Zentral), Südamerika (São Paulo), Asien-Pazifik (Seoul), Asien-Pazifik (Mumbai) und Europa (Paris). Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.	27. September 2018

Änderung	Beschreibung	Datum
Unterstützung für zusätzliche Docker-Flags in Containerdefinitionen hinzugefügt.	Amazon ECS hat Unterstützung für die folgenden Docker-Flags in Containerdefinitionen hinzugefügt: • Systemkontrollen • Interactive • Pseudo-Terminal	17. September 2018
Unterstützung der Authentifizierung in privaten Registern für Amazon ECS mithilfe von AWS Fargate-Aufgaben	Amazon ECS hat die Unterstützung für Fargate-Aufgaben mit privater Registrierungsauthentifizierung mithilfe von AWS Secrets Manager eingeführt. Mit diesem Feature können Sie Ihre Anmeldeinformationen sicher speichern, und verweisen Sie dann auf sie in Ihrer Containerdefinition. In diesem Fall können Ihre Aufgaben private Abbilder verwenden. Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.	10. September 2018
Erweiterung der Regionen für Amazon-ECS-Service Discovery	Amazon-ECS-Service Discovery bietet erweiterte Unterstützung in den Regionen Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), EU (Frankfurt) und Europa (London). Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.	30. August 2018
Unterstützung für geplante Aufgaben mit Fargate-Aufgaben	Amazon ECS unterstützt jetzt geplante Aufgaben mit dem Starttyp Fargate.	28. August 2018

Änderung	Beschreibung	Datum
Authentifizierung in privaten Registern mithilfe von Support AWS Secrets Manager	Amazon ECS hat die Unterstützung für private Registrierungsauthentifizierungen mithilfe von AWS Secrets Manager eingeführt. Mit diesem Feature können Sie Ihre Anmeldeinformationen sicher speichern, und verweisen Sie dann auf sie in Ihrer Containerdefinition. In diesem Fall können Ihre Aufgaben private Abbilder verwenden. Weitere Informationen finden Sie unter Verwenden von AWS Nicht-Container-Images in Amazon ECS.	16. August 2018
Unterstützung von Docker-Volumes hinzugefügt	Amazon ECS hat die Unterstützung für Docker-Volumes eingeführt. Weitere Informationen finden Sie unter Speicheroptionen für Amazon ECS-Aufgaben.	9. August 2018
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Regionen Europa (Frankfurt), Asien-Pazifik (Singapur) und Asien-Pazifik (Sydney) ausgeweitet. Weitere Informationen finden Sie unter AWS Fargate für Amazon ECS.	19. Juli 2018

Änderung	Beschreibung	Datum
Amazon-ECS-Service Scheduler-Strategien hinzugefügt	Amazon ECS hat das Konzept der Service Scheduler-Strategien eingeführt. Es gibt zwei Strategien für Service-Scheduler: • `REPLICA`: Die Replica-Einplanungsstrategie platziert und die gewünschte Anzahl von Aufgaben in Ihrem Cluster und behält sie bei. Standardmäßig verteilt der Service-Scheduler Aufgaben über Availability Zones. Mit Aufgabenplatzierungsstrategien und -bedingungen können Sie festlegen, wie Aufgaben platziert und beendet werden. Weitere Informationen finden Sie unter Replikat-Strategie. • `DAEMON`: Die Daemon-Einplanungsstrategie stellt genau eine Aufgabe auf jeder aktiven Container-Instance bereit, die alle von Ihnen in Ihrem Cluster angegebenen Platzierungsbedingungen für die Aufgaben erfüllt. Bei Verwendung dieser Strategie ist es nicht erforderlich, eine gewünschte Anzahl von Aufgaben oder eine Aufgabenplatzierungsstrateg ie anzugeben oder Auto-Scaling-Richtlinien zu verwenden. Weitere Informationen finden Sie unter Daemon-Strategie. **Note** Fargate-Aufgaben unterstützen die `DAEMON`-Einplanungsstrategie nicht. .	12. Juni 2018

Änderung	Beschreibung	Datum
Amazon-ECS-Container-Agent 1.18.0	Neue Version des Amazon-ECS-Container-Agenten mit folgender zusätzlicher Funktionalität veröffentlicht: • Das Verhalten des Container-Agenten beim Abrufen von Abbildern kann jetzt mit dem Parameter `ECS_IMAGE_PULL_BEHAVIOR` angepasst werden. Weitere Informationen finden Sie unter Konfiguration des Amazon-ECS-Container-Agenten. Weitere Informationen finden Sie auf amazon-ecs-agent Github.	24. Mai 2018
Unterstützung für die Netzwerkmodi `bridge` und `host` bei der Konfiguration von Service Discovery hinzugefügt	Unterstützung für die Konfiguration von Service Discovery für Amazon-ECS-Services mit Aufgabend efinitionen hinzugefügt, die die Netzwerkmodi `bridge` oder `host` angeben. Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.	22. Mai 2018
Weitere Amazon-ECS-optimierte AMI-Metad atenparameter werden jetzt unterstützt	Unterparameter zum programmgesteuerten Abrufen von ID, Image-Name, Betriebssystem, Container-Agentenversion und Laufzeitversion des Amazon-EC S-optimierten AMI wurden hinzugefügt. Die Metadaten können mit der Systems Manager-Parameter-Store-API abgefragt werden. Weitere Informationen finden Sie unter Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten.	9. Mai 2018
AWS Erweiterung der Region Fargate	AWS Fargate mit Amazon ECS wurde auf die Regionen USA Ost (Ohio), USA West (Oregon) und EU West (Irland) ausgeweitet. Weitere Informationen finden Sie unter AWS Fargatefü r Amazon ECS.	26. April 2018

Änderung	Beschreibung	Datum
Amazon-ECS-optimierter AMI-Metadatenabruf	Die Fähigkeit, Amazon-ECS-optimierte AMI-Metadaten mithilfe der Systems Manager-Parameterspeicher-API programmgesteuert abzurufen, wurde hinzufügt. Weitere Informationen finden Sie unter Abrufen von Amazon ECS-optimierten Linux-AMI-Metadaten.	10. April 2018
AWS Fargate-Plattformversion	Neue Version der AWS Fargate-Plattform veröffentlicht, die Folgendes enthält: • Unterstützung für Überwachen Sie Workloads mithilfe von Amazon ECS-Metadaten hinzugefügt. • Unterstützung für Zustandsprüfung hinzugefügt. • Unterstützung für Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden hinzugefügt Weitere Informationen finden Sie unter Fargate Linux-Plattformversionen für Amazon ECS.	26. März 2018
Amazon-ECS-Service Discovery	Integration in Route 53 wurde hinzugefügt, um die Amazon-ECS-Service Discovery zu unterstützen. Weitere Informationen finden Sie unter Verwenden Sie Service Discovery, um Amazon ECS-Services mit DNS-Namen zu verbinden.	22. März 2018
Unterstützung der Docker-Parameter shm-size und tmpfs	Unterstützung der Docker-Parameter shm-sitze und tmpfs in Amazon-ECS-Aufgabendefinitionen. Weitere Informationen über die aktualisierte ECS CLI-Syntax finden Sie unter Linux-Parameter.	20. März 2018

Änderung	Beschreibung	Datum
Container-Zustands prüfungen	Unterstützung von Docker-Zustandsprüfungen in Container-Definitionen hinzugefügt. Weitere Informati onen finden Sie unter Zustandsprüfung.	8. März 2018
AWS Fargate	Übersicht für Amazon ECS mit AWS Fargate hinzugefügt. Weitere Informationen finden Sie unter AWS Fargatefür Amazon ECS.	22. Februar 2018
Amazon-ECS-Endpunkt für Aufgabenmetadaten	Ab Version 1.17.0 des Amazon-ECS-Contain eragenten sind für Aufgaben, die den `awsvpc`-Netzwerkmodus für einen HTTP-Endpunkt verwenden, der vom Amazon-ECS-Containeragenten bereitgestellt wird, verschiedene Aufgabenmetadaten und Docker-Statistiken verfügbar. Weitere Informationen finden Sie unter Überwachen Sie Workloads mithilfe von Amazon ECS-Metadaten.	8. Februar 2018
Auto Scaling von Amazon-ECS-Service s mithilfe von Ziel-Nach verfolgungsrichtlinien	Unterstützung des Auto Scaling des ECS-Service mithilfe von Ziel-Nachverfolgungsrichtlinien in der Amazon-ECS-Konsole wurden hinzugefügt. Weitere Informationen finden Sie unter Skalieren Sie Ihren Amazon ECS-Service mithilfe eines Zielmetrikwerts. Vorheriges Tutorial für die schrittweise Abskalierung im ECS-Assistenten für die erste Ausführung entfernt. Dies wurde durch das neue Tutorial für die Ziel-Verf olgung ersetzt.	8. Februar 2018
Unterstützung von Docker 17.09	Unterstützung für Docker 17.09 hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	18. Januar 2018

Änderung	Beschreibung	Datum
Neues Verhalten des Service-Schedulers	Informationen über das Verhalten für Service-A ufgaben aktualisiert, die nicht starten. Dokumentierte neue Serviceereignismeldung, die ausgelöst wird, wenn bei einer Serviceaufgabe aufeinanderfolgende Fehler auftreten.	11. Januar 2018
Wartezeit für Initialis ierung der Elastic Load Balancing-Zustands prüfung	Möglichkeit hinzugefügt, eine Wartezeit für Zustandsp rüfungen anzugeben.	27. Dezember 2017
CPU und Speicher auf Aufgabenebene	Unterstützung für die Angabe von CPU und Speicher auf Aufgabenebene in Aufgabendefinitionen hinzugefü gt. Weitere Informationen finden Sie unter TaskDefin ition.	12. Dezember 2017
Aufgabenausführung srolle	Der Amazon-ECS-Container-Agent sendet Aufrufe an die Amazon-ECS-API-Aktionen in Ihrem Namen, deshalb benötigt er eine IAM-Richtlinie und - Rolle, damit der Service weiß, dass der Agent zu Ihnen gehört. Die folgenden Aktionen sind von der Aufgabenausführungsrolle abgedeckt: • Aufrufe an Amazon ECR, um das Container-Image abzurufen • Ruft CloudWatch zum Speichern von Container-Anwendungsprotokollen auf Weitere Informationen finden Sie unter IAM-Rolle für die Amazon-ECS-Aufgabenausführung.	7. Dezember 2017
Windows-Container unterstützen GA	Unterstützung für Windows Server 2016-Container hinzugefügt. Weitere Informationen finden Sie unter Amazon-ECS-optimierte AMI-Varianten.	5. Dezember 2017

Änderung	Beschreibung	Datum
AWS Fargate, Georgia	Unterstützung für das Starten von Amazon-ECS-Services mit dem Starttyp Fargate hinzugefügt. Weitere Informationen finden Sie unter Amazon-ECS-Starttypen.	29. November 2017
Amazon-ECS-Namensänderung	Amazon Elastic Container Service wurde umbenannt (zuvor Amazon EC2 Container Service).	21. November 2017
Aufgabenvernetzung	Die Aufgabenvernetzungsfeatures, die durch den Netzwerkmodus awsvpc bereitgestellt werden, geben Amazon-ECS-Aufgaben dieselben Netzwerkeigenschaften wie Amazon EC2-Instances. Wenn Sie den awsvpc-Netzwerkmodus in Ihren Aufgabendefinitionen verwenden, erhält jede Aufgabe, die von dieser Aufgabendefinition gestartet wird, ihre eigene Elastic-Network-Schnittstelle, eine primäre private IP-Adresse und einen internen DNS-Hostnamen. Die Aufgabe-Netzwerkfeatures vereinfacht das Containernetzwerk und gibt Ihnen mehr Kontrolle darüber, wie Anwendungen in Containern miteinander und mit anderen Services in Ihren VPCs kommunizieren. Weitere Informationen finden Sie unter Netzwerkoptionen für Amazon ECS-Aufgaben für den EC2-Starttyp.	14. November 2017
Amazon ECS Container-Metadaten	Amazon-ECS-Container können jetzt auf Metadaten zugreifen, wie ihren Docker-Container oder die Image-ID, die Netzwerkkonfiguration oder Amazon ARNs. Weitere Informationen finden Sie unter Amazon ECS Container-Metadatendatei.	2. November 2017
Unterstützung von Docker 17.06	Unterstützung für Docker 17.06 wurde hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	2. November 2017

Änderung	Beschreibung	Datum
Unterstützung von Docker-Flags: device und init	Unterstützung der device- und init-Features von Docker in Aufgabendefinitionen mit dem Parameter `LinuxParameters` (`devices` und `initProcessEnabled`) hinzugefügt. Weitere Informationen finden Sie unter LinuxParameters.	2. November 2017
Unterstützung von Docker-Flags: cap-add und cap-drop	Unterstützung der cap-add- und cap-drop-Features von Docker in Aufgabendefinitionen mit dem Parameter `LinuxParameters` (`capabilities`) hinzugefügt. Weitere Informationen finden Sie unter LinuxParameters.	22. September 2017
Unterstützung von Network Load Balancer	Amazon ECS hat die Unterstützung für Network Load Balancers in der Amazon-ECS-Konsole hinzugefügt.	7. September 2017
RunTask überschreibt	Unterstützung für das Überschreiben von Aufgabendefinition bei der Ausführung einer Aufgabe hinzugefügt. Auf diese Weise können Sie eine Aufgabe ausführen, während Sie eine Aufgabendefinition ändern, ohne dass Sie eine Revision der neuen Aufgabendefinition erstellen müssen. Weitere Informationen finden Sie unter Eine Anwendung als Amazon ECS-Aufgabe ausführen.	27. Juni 2017
Geplante Amazon-ECS-Tasks	Unterstützung für die Planung von Aufgaben über cron hinzugefügt.	7. Juni 2017
Spot-Instances in der Amazon-ECS-Konsole	Unterstützung für das Erstellen von Spot-Instance-Container-Instances in der Amazon-ECS-Konsole hinzugefügt. Weitere Informationen finden Sie unter Starten einer Amazon ECS Linux-Container-Instance.	6. Juni 2017
Amazon SNS-Benachrichtigung für neue Amazon-ECS-optimierte AMI-Versionen	Möglichkeit hinzugefügt, SNS-Benachrichtigungen über neue Amazon-ECS-optimierte AMI-Versionen zu abonnieren.	23. März 2017

Änderung	Beschreibung	Datum
Microservices und Batch-Aufträge	Dokumentation für zwei häufige Anwendungsfälle für Amazon ECS hinzugefügt: Microservices und Batch-Aufträge. Weitere Informationen finden Sie unter Verwandte Informationen zu Amazon ECS.	Februar 2017
Container-Instance-Ausgleich	Unterstützung für den Container-Instance-Ausgleich hinzugefügt, womit eine Methode eingeführt wird, Container-Instances aus einem Cluster zu entfernen. Weitere Informationen finden Sie unter Entleeren von Amazon ECS-Container-Instances.	24. Januar 2017
Unterstützung von Docker 1.12	Unterstützung für Docker 1.12 wurde hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	24. Januar 2017
Neue Aufgabenplatzierungsstrategien	Unterstützung für Aufgabenplatzierungsstrategien hinzugefügt: Attribut-basierte Platzierung, Binpack, Availability Zone-Verteilung und eine pro Host. Weitere Informationen finden Sie unter Verwenden Sie Strategien, um die Amazon ECS-Aufgabenverteilung zu definieren.	29. Dezember 2016
Unterstützung von Windows-Containern in der Beta	Unterstützung für Windows Server 2016-Container hinzugefügt (Beta). Weitere Informationen finden Sie unter Amazon-ECS-optimierte AMI-Varianten.	20. Dezember 2016
Unterstützung von Blox OSS	Unterstützung für Blox OSS hinzugefügt, womit benutzerdefinierten Aufgaben-Scheduler möglich sind. Weitere Informationen finden Sie unter Planen Sie Ihre Container auf Amazon ECS.	1. Dezember 2016
Amazon ECS-Event-Stream für CloudWatch Veranstaltungen	Amazon ECS sendet jetzt Änderungen des Container-Instance- und Task-Status an CloudWatch Events. Weitere Informationen finden Sie unter Automatisieren Sie Antworten auf Amazon ECS-Fehler mit EventBridge.	21. November 2016

Änderung	Beschreibung	Datum
Amazon ECS-Container-Protokollierung in CloudWatch Logs	Unterstützung für den awslogs-Treiber hinzugefügt, um Container-Protokollstreams an Logs zu CloudWatch senden. Weitere Informationen finden Sie unter Amazon ECS-Protokolle senden an CloudWatch .	12. September 2016
Amazon-ECS-Services mit Elastic Load Balancing Unterstützung für dynamische Ports	Unterstützung für Load Balancer hinzugefügt, um mehrere Instance: Port-Kombinationen pro Listener zu unterstützen, womit die Flexibilität für Container erhöht wird. Jetzt können Sie Docker dynamisch den Host-Port des Containers definieren lassen, und der ECS-Scheduler registriert den Instance:Port bei dem Load Balancer. Weitere Informationen finden Sie unter Verwenden Sie Load Balancing, um den Amazon ECS-Serviceverkehr zu verteilen.	11. August 2016
IAM-Rollen für Amazon-ECS-Aufgaben	Unterstützung für die Zuordnung von IAM-Rollen zu einer Aufgabe hinzugefügt. Damit sind genauer abgestimmte Berechtigungen für Container möglich, anders als bei Verwendung einer einzigen Rolle für eine ganze Container-Instance. Weitere Informationen finden Sie unter IAM-Rolle für Amazon ECS-Aufgaben.	13. Juli 2016
Unterstützung von Docker 1.11	Unterstützung für Docker 1.11 wurde hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	31. Mai 2016
Auto Scaling für Aufgaben	Amazon ECS hat eine Unterstützung für ein Auto Scaling Ihrer von einem Service ausgeführten Aufgaben hinzugefügt. Weitere Informationen finden Sie unter Skalieren Sie Ihren Amazon ECS-Service automatisch.	18. Mai 2016
Filtern der Aufgabendefinition nach Aufgabenfamilie	Unterstützung für das Filtern einer Liste von Aufgabendefinition basierend auf der Aufgabendefinitionsfamilie hinzugefügt. Weitere Informationen finden Sie unter. ListTaskDefinitions	17. Mai 2016

Änderung	Beschreibung	Datum
Docker-Container- und Amazon-ECS-Agent-Protokollierung	Amazon ECS hat die Möglichkeit hinzugefügt, ECS-Agenten- und Docker-Container-Protokolle von Container-Instances an CloudWatch Logs zu senden, um die Problembehandlung zu vereinfachen.	5. Mai 2016
Die ECS-optimierte AMI unterstützt jetzt Amazon Linux 2016.03.	Das ECS-optimierte AMI hat Unterstützung für Amazon Linux 2016.03 hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	5. April 2016
Unterstützung von Docker 1.9	Unterstützung für Docker 1.9 wurde hinzugefügt. Weitere Informationen finden Sie unter Amazon ECS-optimierte Linux-AMIs.	22. Dezember 2015
CloudWatch Metriken für Cluster-CPU und Speicherreservierung	Amazon ECS hat benutzerdefinierte CloudWatch Metriken für CPU- und Speicherreservierung hinzugefügt.	22. Dezember 2015
Neue Wahrnehmung bei der ersten Amazon-ECS-Ausführung	Der Wahrnehmung bei der ersten Ausführung der Amazon-ECS-Konsole wurde die Zero-Click-Rollenerstellung hinzugefügt.	23. November 2015
Aufgabenplatzierung über mehrere Availability Zones	Der Amazon-ECS-Service-Scheduler hat eine Unterstützung für die Aufgabenplatzierung über mehrere Availability Zones hinzugefügt.	8. Oktober 2015
CloudWatch Metriken für Amazon ECS-Cluster und -Services	Amazon ECS hat benutzerdefinierte CloudWatch Metriken für die CPU- und Speicherauslastung für jede Container-Instance, jeden Service und jede Aufgabendefinitionsfamilie in einem Cluster hinzugefügt. Diese neuen Metriken können verwendet werden, um Container-Instances in einem Cluster mithilfe von Auto Scaling-Gruppen zu skalieren oder benutzerdefinierte CloudWatch Alarme zu erstellen.	17. August 2015

Änderung	Beschreibung	Datum
Unterstützung von UDP-Ports	Unterstützung für UDP-Ports in Aufgabendefinitionen hinzugefügt.	7. Juli 2015
Überschreiben von Umgebungsvariablen	Unterstützung für deregisterTaskDefinition und das Überschreiben von Umgebungsvariablen für RunTask hinzugefügt.	18. Juni 2015
Automatisierung von Amazon ECS Agent-Updates	Möglichkeit hinzugefügt, die ECS-Agent-Version anzuzeigen, die auf einer Container-Instance ausgeführt wird. Kann den ECS-Agenten auch über das AWS Management Console AWS CLI, und SDK aktualisieren.	11. Juni 2015
Amazon-ECS-Service-Scheduler und Elastic Load Balancing Integration	Möglichkeit hinzugefügt, einen Service zu definieren und diesen Services einem Elastic Load Balancing-Load Balancer zuzuordnen.	9. April 2015
Amazon ECS GA	Allgemeine Verfügbarkeit von Amazon ECS in den Regionen USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Tokio) und Europa (Irland).	9. April 2015

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.